Modélisation 2D 1/2 hiérarchique basée sur les cartes planaires : réalisation et évaluation d'une interface graphique utilisant cette modélisation

Nahed Abdelfattah

ÉCOLE NATIONALE SUPÉRIEURE
DES MINES DE SAINT-ETIENNE

N° d'ordre : 105 ID

THÈSE

Présentée par

Abdelfattah NAHED

pour obtenir le titre de

DOCTEUR

DE L'UNIVERSITÉ DE SAINT-ETIENNE
ET DE L'ÉCOLE NATIONALE SUPÉRIEURE DES MINES DE SAINT-ETIENNE

(Spécialité : Informatique)

Modélisation $2D^{1/2}$ hiérarchique basée sur les cartes planaires; réalisation et évaluation d'une interface graphique utilisant cette modélisation

soutenue à SAINT-ETIENNE le 28 Février 1994

Composition du Jury :

Président :
M. B. PÉROCHE

Rapporteurs :
M. D. ARQUÈS
M. J. P. BRAQUELAIRE

Examinateurs :
M. M. NANARD
M. J. P. SCHON
M. E. TOSAN

ÉCOLE NATIONALE SUPÉRIEURE
DES MINES DE SAINT-ETIENNE

N° d'ordre : 105 ID

THÈSE

Présentée par

Abdelfattah NAHED

pour obtenir le titre de

DOCTEUR

DE L'UNIVERSITÉ DE SAINT-ETIENNE
ET DE L'ÉCOLE NATIONALE SUPÉRIEURE DES MINES DE SAINT-ETIENNE

(Spécialité : Informatique)

Modélisation $2D^{1/2}$ hiérarchique basée sur les cartes planaires; réalisation et évaluation d'une interface graphique utilisant cette modélisation

soutenue à SAINT-ETIENNE le 28 Février 1994

Composition du Jury :

Président :
M. B. PÉROCHE

Rapporteurs :
M. D. ARQUÈS
M. J. P. BRAQUELAIRE

Examinateurs :
M. M. NANARD
M. J. P. SCHON
M. E. TOSAN

Remerciements

Je tiens beaucoup à remercier toutes les personnes qui ont contribué directement ou indirectement à la réalisation de ce travail en permettant son déroulement dans un cadre stable et serein. Je pense bien sûr à mon entourage proche, à ma famille et à mes amis qui m'ont supporté, et me supporteront encore, et m'ont encouragé tout le long de mes années de thèse. Je ne m'aventurerai pas à les citer chacun par son nom : une page n'y suffirai pas.

Mes remerciements sont destinés tout spécialement à tous les membres du jury :

M. Didier ARQUÈS, professeur à l'Université de Franche-Comté et M. Jean-Pierre BRAQUELAIRE, professeur à l'Université de Bordeaux I, pour avoir accepté d'être rapporteurs de cette thèse et pour leurs remarques et suggestions qui ont contribué à l'amélioration de ce mémoire.

M. Marc NANARD, professeur à l'Université des Sciences et Techniques du Languedoc.

M. Jean-Paul SCHON, professeur à l'Université Jean Monnet de Saint-Etienne.

M. Eric TOSAN, ingénieur de recherche à l'Université Claude Bernard Lyon I, pour ses suggestions intéressantes concernant le premier chapitre de ce mémoire.

M. Bernard PÉROCHE, professeur à l'Ecole des Mines de Saint-Etienne, pour m'avoir accueilli au sein du département d'informatique appliquée qu'il dirige et pour le temps précieux qu'il a consacré à mon encadrement et aux nombreuses lectures et relectures des différents documents produits tout au long de cette thèse. Je lui en suis très reconnaissant.

Je n'oublie pas toutes les personnes du département d'informatique appliquée qui ont rendu ces années de thèse bien agréables. Je pense tout d'abord à mes chers évaluateurs volontaires : Marie-Line, Christine, Florence, Elisabeth, Michel, Dominique, Antoine, Jean-Michel, Laurent et Jean-Luc, sans lesquels je me demande encore comment j'aurai pu effectuer le travail d'évaluation.

Merci aussi à tous les autres qui n'ont pas pu participer à l'évaluation et qui sont de tout coeur avec moi : Nadine, Annie, Pascale, Suzan, Jori, Clément, Jean-Pierre, Daniel, Jean-Paul, Marc, Rabah, Michel, Samy, François, JiJi, Roland, Jean-François, Philippe, Bernard, Gabriel,

Helmi. Je n'oublie pas les nouveaux Mihaela, Ibrahima, Gauthier, Gilles, Mustapha, Franck, Hakim, Hervé, Nicolas, ni nos autres collègues 'SIMADE' Mme Girard, Mme ZOLD, Mlle MAZER, Mlle Perreard, M. Jullien, M. Beaune, M. Framling, M. Senoune ... notre bibliothéquaire Madame Sayet et mes amis les chimistes.

Je remercie enfin tous les membres du service de reprographie de l'Ecole des Mines de Saint-Etienne sans qui cette thèse ne serait pas entre vos mains cher lecteur.

SOMMAIRE

INTRODUCTION GÉNÉRALE

Le concept de carte planaire, introduit par des mathématiciens [BERG 58], a été développé dans les années 70 par des informaticiens [JACQ 70] et a trouvé de nombreuses applications, notamment en modélisation. Si la notion de carte planaire a eu tant de succès, elle le doit en priorité à sa capacité à modéliser la topologie de scènes assez complexes. La topologie dans ce cadre là s'exprime en terme d'incidence et d'adjacence entre les entités constituantes d'une carte planaire à savoir les sommets, les arêtes et les faces. Outre la topologie, des informations d'ordre géométrique peuvent être attachées aux cartes planaires : coordonnées des sommets, représentation mathématique de la forme des arêtes ...

Certaines applications utilisant le concept carte planaire permettent la prise en compte d'arêtes courbes définies par leurs équations algébriques et leurs sommets de départ et d'arrivée. Ces courbes peuvent être des courbes de Bézier rationnelles, des courbes discrètes ...

Les travaux que nous présentons dans ce document recouvrent deux aspects qui s'inscrivent de façon naturelle dans le cadre d'un projet européen Esprit :

- un aspect théorique comportant une modélisation $2D^{1/2}$ hiérarchique basée sur les cartes planaires. Cette modélisation s'appuie sur une structure de données 'carte planaire' où les arêtes sont représentées graphiquement par des segments de droite,

- un aspect pratique de réalisation : partant d'une interface graphique, à laquelle nous avons contribué au début de la thèse, nous avons effectué une évaluation ergonomique de cette interface et l'avons améliorée pour évoluer vers l'interface graphique qui sera présentée dans le chapitre III.

Nous employons le terme 'carte planaire' pour désigner une structure de données qui code le graphe planaire associé à un lot de segments du plan euclidien. Cette structure de données a été proposée par Michelucci avec un algorithme dit de construction statique qui déduit la topologie 'carte planaire' à partir de la donnée géométrique de segments dans un plan. La principale caractéristique de cette structure est qu'elle permet de représenter à la fois des données topologiques et des données géométriques. De plus, cette structure offre la possiblité d'associer de manière quasi-naturelle un sens sémantique à chacun des constituants

'topologiques' d'une carte planaire : les sommets, les arêtes et les faces. Pour une meilleure efficacité et afin de rendre cette structure mieux adaptée à des applications de type interactif, Ben Amara a proposé un algorithme de construction incrémentale. Il permet la mise à jour de la structure 'carte planaire' suite à des opérations d'ajout ou de suppression d'arêtes et donc sans entière reconstruction. Cela a l'avantage de préserver l'ensemble des attributs sémantiques qui pourraient être attachés aux constituants d'une carte planaire (sommets, arêtes, faces) après de nombreuses interactions entre un utilisateur et une application. Nous tenons à préciser que dans le présent document, nous avons choisi de présenter les diverses structures de données avec la syntaxe du langage C utilisé lors du codage.

Le modèle $2D^{1/2}$ hiérarchique utilise cette structure de données et les algorithmes de construction associés (statique et incrémental). Il permet de modéliser des scènes faites essentiellement d'objets sémantiques représentables par des segments et caractérisés, chacun, par sa classe sémantique et son niveau. Ce modèle permet de modéliser des objets répartis sur des niveaux plans strictement parallèles. Les objets ayant la même classe sématique (bâtiment, circuits électriques, canalisations d'eau ...) et appartenant au même niveau plan sont représentés dans une même carte planaire. La structure de données adoptée pour cela est en quelque sorte un empilement de cartes planaires indépendantes. Le sens global de la scène modélisée est préservé grâce à des structures de connexion qui permettent d'établir des liens entre les objets indépendamment de leur sémantique et de leur niveau. Une classe sémantique dite sémantique de base a été privilégiée par rapport aux autres classes qui sont dites des sémantiques auxiliaires. La carte planaire (de base) associée à la sémantique de base représente une sorte de propriétaire de l'ensemble des objets sémantiques : la sémantique de bâtiment par rapport à celle des circuits d'électricité ou des canalisations d'eau. Une représentation dite tri-dimensionnelle inspirée des techniques de dessin en perspective cavalière permet la visualisation et la manipulation d'objets appartenant à des plans strictement parallèles.

La sémantique de base est également utilisée pour bâtir une hiérarchie de raffinements (détails) de certaines parties de chacun des niveaux de la scène. Les parties d'un niveau sont délimitées par des faces de la sémantique de base. En raison de la superposition de cartes planaires représentant plusieurs sémantiques, à un raffinement des faces de base a été associée une superposition de cartes planaires. Cette hiérarchie de niveaux de détail pose le problème des représentations multiples d'un même objet physique et celui de la prise en compte des objets de sémantique auxiliaire qui sortent d'une face détaillée (superpostion de cartes planaires indépendantes). Pour gérer la cohérence de cette hiérarchie, nous avons imposé des règles (principes) concernant la spécification des détails et introduit la notion d'élément de continuité géométrique. Ces éléments ont pour rôle principal d'assurer une coupe symbolique des objets auxiliaires qui sortent des faces détaillées et d'assurer une continuité géométrique entre la description d'un niveau de détail i+1 (intérieur d'une face) et celle du niveau i (extérieur de la face détaillée).

Nous avons développé une interface graphique validant le modèle $2D^{1/2}$ hiérarchique basé sur la notion de carte planaire. Cette interface offre en plus l'avantage de traiter des dessins à main levée saisis par manipulation directe d'outils (main levée, segment, rectangle et ligne brisée). Des algorithmes dits de pré-traitement de dessins à main levée sont présentés dans le

troisième chapitre de ce document. Ils transforment les dessins saisis en un lot de segments à l'aide de techniques d'échantillonnage, de recalage et d'heuristiques pour l'élimination des ratures. Cette interface graphique a été développée dans le cadre d'un projet européen Esprit II MMI^2 visant le développement d'une interface multi-modes : langue naturelle (français, anglais et espagnol), un langage de commande, le graphique et le gestuel. L'interface graphique constitue l'outil interactif de saisie et de manipulation directe du mode graphique qui comprend également des outils de présentation d'informations (camemberts, histogrammes ...) développés par un des partenaires dans le projet.

L'interface graphique développée a été intégrée dans un prototype présenté dans la semaine des projets Esprit qui a eu lieu en Novembre 1991 à Bruxelles. Ce projet qui a été initié pour une durée de trois années a été prolongé de deux années supplémentaires grâce aux résultats satisfaisants obtenus. Durant ces années supplémentaires, nous avons effectué une évaluation de l'interface graphique développée dans notre laboratoire et nous l'avons exploitée en apportant des améliorations à l'interface évaluée. L'interface améliorée est présentée dans le troisième chapitre de ce documment après une brève présentation du projet MMI^2.

Le plan de la thèse sera le suivant :

le chapitre I contiendra des rappels sur les cartes planaires : cartes combinatoires, structure de données, sémantique et algorithmes de construction,

le chapitre II présentera le modèle 2D$^{1/2}$ hiérarchique basé sur les cartes planaires,

le chapitre III présente la partie réalisation validant le modèle présenté dans le chapitre II, réalisation tenant compte de l'évaluation ergonomique effectuée (cf. chapitre IV). Ce chapitre présente d'abord le projet européen Esprit MMI^2 visant la réalisation d'une interface multimodes; il présente aussi l'ensemble des modes de dialogue pris en compte dans MMI^2,

le chapitre IV contient une étude bibliographique dans le domaine de l'évaluation des interfaces homme-machine. Cette étude s'est concrétisée par l'évaluation du premier prototype de l'interface graphique développée pendant les trois premières années du projet MMI^2. Ce prototype a subi des améliorations suite à l'évaluation et a évolué vers l'interface graphique présentée dans le chapitre III.

Cette thèse se termine par une conclusion et des annexes relatives à l'évaluation (cf. chapitre IV).

CHAPITRE I

LES CARTES PLANAIRES

LE MODÈLE BI-DIMENSIONNEL DES CARTES PLANAIRES

Ce chapitre est consacré à une présentation du modèle bi-dimensionnel des cartes planaires tel qu'il a été développé dans notre laboratoire. Nous employons le terme 'carte planaire' pour désigner une structure de données qui code le graphe planaire associé à un lot de segments du plan euclidien. Cette structure de données a été proposée et utilisée par Michelucci lors du développement d'un logiciel de saisie d'esquisses d'architecture [MICH 84]. La principale caractéristique de cette structure est qu'elle permet de représenter à la fois des données topologiques et des données géométriques. Plus précisément, les algorithmes de construction que nous allons présenter dans les sections suivantes permettent de déduire la topologie à partir de la géométrie. De plus, cette structure offre la possiblité d'associer de manière quasi-naturelle un sens sémantique à chacun des constituants 'topologiques' d'une carte planaire : les sommets, les arêtes et les faces. La richesse de cette structure de données a fait que nous l'avons choisie pour bâtir le modèle $2D^{1/2}$ hiérarchique que nous présenterons dans le deuxième chapitre.

Dans ce qui suit, nous ferons d'abord un rappel de la définition du concept carte planaire et le lien entre la topologie et la géométrie dans ce concept. Ensuite, nous présenterons la structure de données sous-jacente au modèle bi-dimensionnel basé sur les cartes planaires; puis, nous établirons le lien entre les cartes planaires et le sens sémantique par quelques exemples. Enfin, nous présenterons les algorithmes de construction des cartes planaires et nous concluerons.

1. RAPPELS SUR LES CARTES PLANAIRES

La plupart des études et publications se reliant au concept de carte planaire sont centrées sur les aspects topologiques qui ont comme fondement mathématique la théorie des graphes.

On appelle **carte planaire** ou graphe planaire topologique, selon les termes de Berge [BERG 58], [BERG 83] et de Roy [ROY 69], **une représentation plane d'un graphe planaire**.Busacker et Saaty [BUSA 65] se sont également intéressés aux graphes planaires et à leurs propriétés.

La définition suivante des graphes planaires est extraite de [BERG 83] :

On dit qu'un graphe G est planaire s'il est possible de le représenter sur un plan de sorte que les sommets soient des points distincts, les arêtes des courbes simples, et que deux arêtes ne se rencontrent pas en dehors de leurs extrémités.

1.1. Les cartes combinatoires : quelques définitions

D'après [CORI 84], le plongement d'un graphe dans une surface est considéré comme un couple (σ, α) formé d'une permutation σ et d'une involution sans point fixe α qui engendrent un groupe opérant transitivement sur un ensemble fini de brins. Ce couple est souvent appelé carte combinatoire.

Orbites, arêtes, sommets et faces

Soient $A \subseteq B$ un ensemble de brins et τ une permutation de B, on appelle **orbite** de A par rapport à τ la suite $<\tau^p(b)>_{b\in A,\ p\geq 0}$ noté τ^*A. Elle définit l'ensemble des brins de B accessibles par τ à partir de ceux de A.

$a = \alpha^*b = <b, \alpha(b)>$ où $b \in B$ est appelé **arête** de la carte G; b et $\alpha(b)$ sont dits opposés. Pour un brin b, l'opposé de b est indifféremment noté $\alpha(b)$ ou -b.

$s= \sigma^*b = <b, \sigma(b), \sigma^2(b),.....,\sigma^{k-1}(b)>$ où k est le plus petit entier naturel non nul tel que $\sigma^k(b)= b$ est appelé **sommet** de G. Les brins b et $\sigma(b)$ sont dits adjacents.

L'arête $a = \alpha^*b = <b, \alpha(b)>$ est dite **incidente** en chacun de ses sommets $s1=\sigma^*b$ et $s2=\sigma^*[\alpha(b)]$.

Tous les brins $\sigma^j(b)$ appartenant à l'orbite définissant le sommet $s=\sigma^*b$ sont dits **incidents** au sommet s.

On appelle **face directe** la suite finie $<b_{i1},b_{i2}..... b_{in}>$ de G tel que : $\forall$ k,$\exists$j, $b_{ik}=\sigma^{-1}o\ \alpha\ (b_{ij})$.

1.2. Les cartes planaires et la géométrie

En général, la représentation graphique des arêtes est réalisée par des segments. On parle alors de "Planar Straight-Line Graphs" PSLG [MULL 78]. Mais, certaines applications permettent la prise en compte d'arêtes courbes définies par leurs équations algébriques et leurs sommets de départ et d'arrivée. Ces courbes peuvent être des courbes de Bézier rationnelles [GANG 89], des courbes discrètes [BRAQ 90], [DOME 91] ...

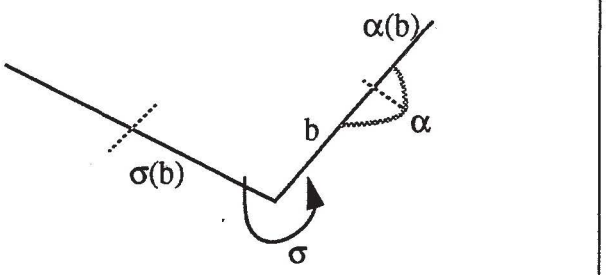


Figure 1: les brins et les applications σ et α

Nous nous sommes limité aux cartes planaires du type PSLG. Comme nous déduisons la topologie 'carte planaire' à partir de segments du plan, le plongement des cartes planaires dans le plan euclidien s'exprime de façon assez simple :

- à chaque sommet de la carte planaire nous associons de façon bijective un seul point du plan euclidien $\Re^2$ ramené à un repère orthonormé (O, i, j),

- chaque arête peut alors être caractérisée par les points A et B images de ses deux sommets dans le plan euclidien; en cas de besoin, les paramètres a, b et c de l'équation cartésienne de la droite supportant le segment représentant l'arête peuvent être déduits de A et de B.

Par abus de langage, nous utiliserons indifféremment les dénominations segment (géométrie) ou arête (topologie). De même, nous utiliserons le terme sommet pour désigner un point du plan euclidien représentant un sommet d'une carte planaire. Du point de vue géométrique et graphique, un brin est une demi-arête. Schématiquement, un brin est un couple formé d'une arête et d'un de ses deux sommets.

Ordre sur les sommets : x-y-ordre

L'ordre lexicographique définit un ordre total dans le plan euclidien. Nous appellerons cet ordre le x-y-ordre. Il classe les sommets par abscisses croissantes; à abscisses égales, les sommets sont classés par ordonnées croissantes. Cet ordre permet de différencier les deux sommets d'une arête : son premier sommet qui est le plus petit des deux par rapport à l'ordre lexicographique, et son second sommet. Une arête étant faite d'une paire de brins, nous distinguons également son premier brin, qui a pour sommet le premier sommet de l'arête, de son second brin qui a pour sommet le second sommet de l'arête. Ainsi, nous dirons qu'une arête **naît** en son premier sommet et **meurt** en son second sommet.

Ordre sur les brins d'un même sommet : σ-ordre

Nous pouvons associer de façon bijective une demi-droite à chaque brin d'une carte planaire. Les demi-droites associées aux brins d'un même sommet S concourent en ce sommet. Nous ordonnons ces demi-droites à l'aide de l'ordre total correspondant à l'angle $\sigma \in]0,2\pi]$ qu'elles font avec la demi-droite verticale descendante à partir de S. Le sens positif de σ est le sens trigonométrique. Cet ordre total sur les demi-droites permet de définir un ordre total sur les brins d'un même sommet. Nous nommons cet ordre le σ-ordre.

Premier brin d'une face - face interne/face externe

Nous avons vu précédemment qu'une face est définie par une suite finie de brins $f=\{b_{i1},b_{i2}..... b_{in}\}$. L'ensemble Sb des sommets associés aux brins de f est fini. Soit s le plus petit sommet de Sb selon le x-y-ordre. Le premier brin de la face f est le plus petit, selon le σ-ordre, parmi les brins incidents au sommet s et qui appartiennent à f.

Etant donné un brin b, son orbite par rapport à l'application $\sigma^{-1}o\ \alpha$ permet de définir une face directe. Si le parcours de cette face à partir de b, par applications successives de $\sigma^{-1}o\ \alpha$, se fait dans le sens des aiguilles d'une montre, alors elle est dite **face externe**; sinon la face est dite **face interne**.

Par abus de langage, une **face interne** sera appelée **face** dans la suite de ce document.

Propriétés

♦ Si *f* est une face interne, il existe $b \in f$ tel que $\alpha(b) \notin f$.

♦ Une face interne *f* est orientée dans le même sens que σ, c'est-à-dire dans le sens trigonométrique, ce qui équivaut à dire que lors du parcours de la face, son intérieur se trouve à gauche.

♦ Une composante connexe d'une carte planaire comporte une et une seule face externe.

♦ Une arête <b,α(b)> appartient soit à deux faces internes distinctes ou confondues, soit à une face interne et à une face externe, soit à une et une seule face externe.

Arête pendante - Arête isolée

Une arête a= <b, α(b)> est dite **pendante** si b et α(b) appartiennent à la même face (interne ou externe).

Une arête a= <b, α(b)> est dite **isolée** si elle est pendante et si σ(b)=b et σ(α(b))=α(b).

Exemple de carte

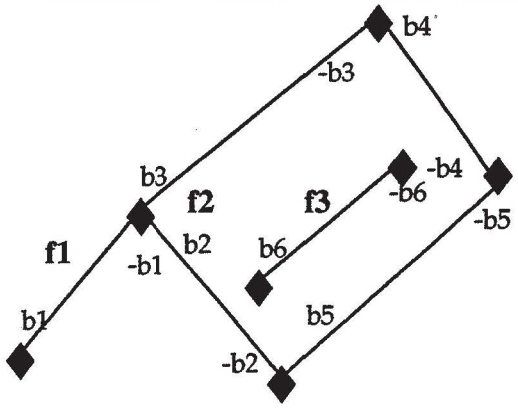


Figure 2 : une carte planaire

Considérons la carte de la figure 2 :

arêtes :

$a_1=<b_1,-b_1>$; $a_2=<b_2,-b_2>$; $a_3=<b_3,-b_3>$;
$a_4=<b_4,-b_4>$; $a_5=<b_5,-b_5>$; $a_6=<b_6,-b_6>$;
a1 est pendante; a6 est isolée

sommets :

$s_1=<b_1>$; $s_2=<b_2,b_3,-b_1>$; $s_3=<b_6>$; $s_4=<b_5,-b_2>$;
$s_5=<b_4,-b_3>$; $s_6=<-b_6>$; $s_7=<-b_4,-b_5>$;

faces :

$f_1=<b_1,b_3,b_4,-b_5,-b_2,-b_1>$ est externe.
$f_2=<b_2,b_5,-b_4,-b_3>$ est interne;
$f_3=<b_6,-b_6>$ est externe;

2. LA STRUCTURE DE DONNEES "CARTE PLANAIRE"

Dans la littérature, plusieurs types de structures de données ont été proposées pour implanter le concept de carte planaire. Parmi ces structures nous distinguons :

- les structures utilisant directement les permutations σ et α des cartes combinatoires [BRAQ 91], [LIEN 88],
- la structure d'arête aîlée de Baumgart [BAUM 72], [BAUM 75],
- la liste doublement chaînée d'arêtes DCEL de Muller et Preparata [MULL 78],
- la structure 4-arête de Guibas et Stolfi [GUIB 85],
- les hypergraphes d'adjacence de faces [ANSA 85],
- la structure de triangle aîlé [PAOL 89], [PAOL 93],
- l'arbre quaternaire exact ou arbre quaternaire non minimal [AYAL 85],
- les structures de données relationnelles envisagées par Dufour [DUFO 88].

La structure de données la plus référencée dans ce domaine est celle de l'arête aîlée. Elle semble se prêter assez bien aux mécanismes de la programmation procédurale et à la gestion de données dynamique. Nous avons opté pour une structure de données qui s'inspire de la structure d'arête aîlée [BAUM 72] ou de la liste doublement chaînée d'arêtes [MULL 78]. Par abus de langage, nous appelons cette structure carte planaire. Cette structure a l'avantage de supporter à la fois la topologie et la géométrie et d'offrir un bon support pour la sémantique.

La structure de données carte planaire se compose de deux parties : la carte planaire locale (**CP_locale**), et la carte planaire globale (**CP_globale**) :

- la **CP_locale** est la partie qui s'inspire de la structure d'arête aîlée ou de la liste doublement chaînée d'arêtes.

```
struct sommet
    {
        struct sommet *suivant;
        struct sommet *precedent;
        struct arete *incidente;
        int cote_incidente;
        int x,y;
        enum_type_som naissance;
        ...
    };
```

```
struct arete
        {
                struct brin demi[2];
                ...
        };
struct brin
        {
                struct sommet *som;
                struct arete *voisine;
                int cote_voisine;
                struct face *face;
                ...
        };
```

La **CP_locale** contient des informations locales à chaque sommet; parmi ces informations nous codons le x-y-ordre local. C'est à dire que chaque enregistrement de *Sommet* permet de passer au sommet précédent et au sommet suivant, selon le x-y-ordre, par l'intermédiaire de pointeurs sur des structures de type *Sommet*. En ce qui concerne les données géométriques, elles se limitent à prévoir les coordonnées cartésiennes x et y d'un point dans le plan euclidien parmi les champs de l'enregistrement *Sommet*. De plus, chaque enregistrement de *Sommet* contient l'adresse de son premier brin selon le σ-ordre : dans une implémentation de type C ou Pascal, où une *Arête* peut être représentée par un tableau de brins à deux éléments, l'adresse d'un brin est complètement déterminée par l'adresse de l'arête qui le porte et un indice prenant les valeurs 0 (premier brin de l'arête) ou 1 (deuxième brin de l'arête). Le σ-ordre autour d'un sommet est implanté comme suit : chaque enregistrement de type *Brin* contient l'adresse de son successeur selon le σ-ordre, le dernier d'entre eux pointant sur le premier. Ainsi, le σ-ordre est implanté par une liste bouclée de brins.

- la **CP_globale** complète la CP_locale dans le sens où elle détermine chacune des faces de la carte planaire et contient des informations globales : contiguïté et inclusions entre faces. Etant donné un brin b, son orbite par rapport à l'application $\sigma^{-1} \circ \alpha$ permet de définir une face directe. Le parcours de cette face se fait en itérant le processus suivant : passer au brin opposé α(b) de la même arête, puis recommencer avec le brin juste inférieur à α(b) pour le σ-ordre autour du sommet de α(b). Ce parcours se fait donc à l'aide de la CP_locale qui est en fait une forme implicite de la CP_globale. La CP_globale est explicitée par la mise en place d' enregistrements appelés *faces* caractérisés par leur premier brin (cf. section 1.2. de ce chapitre) et leur type. Le type d'une face est interne ou externe selon qu'il est parcouru, à partir de son premier brin, suivant le sens trigonométrique ou le sens opposé.

Contrairement à la CP_locale qui définit le σ-ordre autour d'un sommet, la CP_globale permet de matérialiser les faces (ou contours) d'une carte planaire en marquant sur chaque brin la face qui l'emprunte. De plus, la CP_globale contient l'organisation des faces sous forme d'un

arbre qui représente l'ordre naturel induit par les inclusions entre faces. C'est un arbre généalogique binarisé dans lequel chaque face pointe sur sa face parente, sur sa face fille aînée, et sur sa face soeur cadette. Ceci suppose l'existence d'un ordre total entre les fils d'une même face. Cet ordre est défini par le x-y-ordre des sommets de leurs brins de naissance (premier brin); dans le cas où deux faces naissent au même sommet (leurs brins de naissance partent du même sommet) leur ordre est celui du σ-ordre de leurs brins de naissance autour de ce sommet. L'arbre d'inclusion des faces est représenté par la structure de données qui suit.

```
struct face
    {
        struct face *parente;
        struct face *soeur;
        struct face *fille_ainee;
        struct arete *depart;
        enum_sens sens_face;
    };
```

Dans les paragraphes relatifs à la construction d'une carte planaire, nous verrons comment se fait le marquage de chaque brin de la carte planaire par la face qui l'emprunte, comment se fait la distinction entre une face interne et une face externe, à partir de quelles données nous décidons qu'une face est la fille d'une autre ...

3. DE LA SEMANTIQUE DANS LES CARTES PLANAIRES

La structure carte planaire étant constituée de sommets, d'arêtes et de faces, il est aisé d'associer une sémantique à chacune de ces entités. Le logiciel de saisie d'esquisses d'architecture développé par Michelucci et présenté dans [MICH 84] introduit une sémantique dans les cartes planaires en autorisant des désignations et instrumentations du type : épaississement de certaines arêtes, coloriage de zones, ajout d'ouvertures pour les portes et les fenêtres ... Ce genre d'information est souvent géré par une liste de couples (attribut, valeur) proche du concept 'liste de propriétés' de LISP. Ces listes de propriétés sont utilisées par les différents logiciels d'application des cartes planaires [MOIS 84], [COQU 84], [ROMM 87] ... développés dans notre laboratoire.

En fait, aucune application ne peut être envisagée sans sémantique implicite ou explicite. Dans le cas explicite, la sémantique se trouve contenue dans les outils de l'interface de l'application. Les auteurs de [CHEN 92] soulignent que l'information d'ordre sémantique contenue dans les notations utilisées dans le domaine des graphes est spécifique à chaque application. Ils fournissent deux exemples de graphes : le premier (cf. figure 3 a), n'a de sens que pour les programmeurs et représente une indésirable boucle infinie; le second (cf. figure 3 b) représente une transmittance C/R=G1/(1+G1G2). Nous remarquons sur ces deux exemples qu'une orientation a été associée aux arêtes, et que les sommets possèdent une sémantique représentée par une boîte rectangulaire contenant un texte, un losange, un point de jonction, une icône particulière ...

En ce qui concerne l'application que nous présenterons dans le chapitre III de ce document, une carte planaire représentera par exemple le dessin de bâtiment (le gros oeuvre) d'un étage. L'association d'une sémantique à cet étage consiste à créer un objet dont la classe sera *Niveau*. A chaque arête, nous associerons un objet de type (ou classe) *Mur*. A chaque face interne (communément appelée face), nous associerons un objet de type *Pièce* ou *Couloir*. Le tracé et l'intérieur de chaque face sont déterminés par la topologie qui fournit pour chaque face l'ensemble des brins qui la constituent... De même que pour les exemples de la figure 3, certains sommets ont une sémantique dans notre application et une icône est placée en leur image dans le plan euclidien.

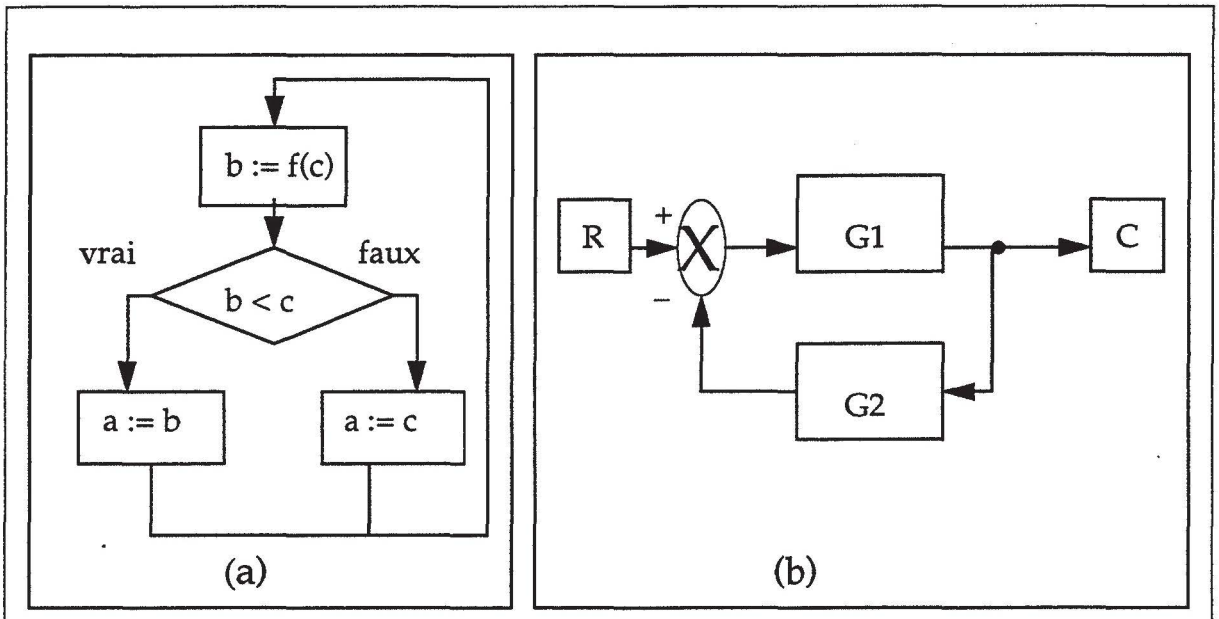


Figure 3 : exemples montrant la dépendance de la sémantique par rapport à l'application

4. ALGORITHMES DE CONSTRUCTION D'UNE CARTE PLANAIRE

Nous pouvons envisager deux manières de construire une carte planaire :

- soit on dispose d'un ensemble de segments non structurés et la carte planaire doit être construite ex nihilo à partir de ces données. Nous parlerons dans ce cas de construction statique.

- soit on dispose d'une carte planaire et on veut rajouter et/ou supprimer un ou plusieurs segments à cette carte. On parlera alors de construction incrémentale.

Dans les sections qui suivent, nous décrirons successivement les algorithmes de construction statique et incrémentale des cartes planaires. Nous utiliserons les applications α et σ qui opèrent sur des brins, σ définissant le σ-ordre autour d'un sommet et α permettant d'accéder à partir d'un bout d'arête (brin) à l'autre bout.

4.1. La construction statique des cartes planaires

Au départ, nous disposons d'un ensemble de segments non structurés. Les données sont de nature géométrique : nous connaissons les coordonnées des deux extrémités de chaque segment dans le plan euclidien. L'ensemble de ces segments doit être traité pour en faire une représentation plane d'un graphe planaire : deux arêtes distinctes ne s'intersectent qu'en un sommet du graphe. Une fois les segments et sommets clairement définis en tant qu'entités géométriques (les points d'intersection entre les segments initiaux étant convertis en sommets), nous devons en déduire la topologie que nous appelons carte planaire faite des deux structures CP_locale et CP_globale.

Dans ce qui suit nous décrirons le processus de construction statique de la structure de données carte planaire. Cette construction se déroule en trois étapes principales : la première étape est une initialisation, la deuxième étape consiste en la mise en place de la structure CP_locale; enfin la troisième étape explicite la CP_globale à partir de la CP_locale.

4.1.1. Initialisation de la carte planaire

Nous créons une structure de données locale à partir de l'ensemble des segments de départ (cf. figure 4). Notons qu'en général cette structure n'est pas cohérente puisqu'elle ignore les intersections entre segments initiaux. Lors de la construction de cette structure, nous confondons les sommets ayant les mêmes coordonnées et les arêtes ayant le même sommet de naissance (premier sommet) et le même sommet de mort (deuxième sommet).

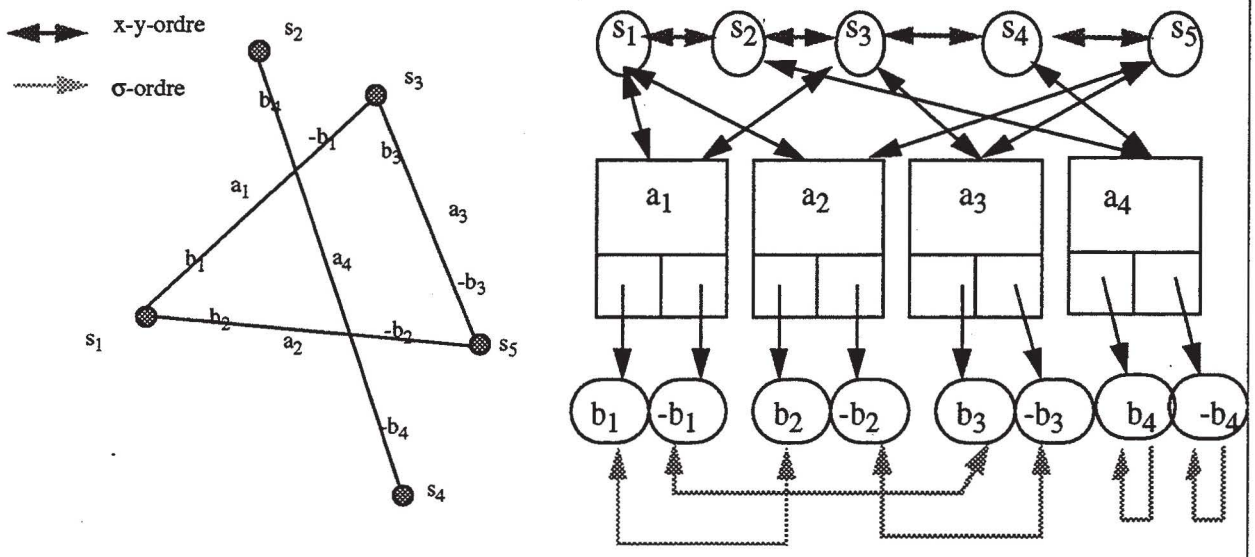


Figure 4 : structure de données construite par l'étape d'initialisation

Chaque sommet s_i est caractérisé par deux entiers x et y (cf. section 2.). Ils correspondent aux coordonnées pixels dans un repère écran (axe des abscisses horizontal orienté vers la droite; axe des ordonnées vertical orienté vers le haut) du point image du sommet dans l'écran.

4.1.2. Construction de la carte planaire locale CP_locale

Après l'étape d'initialisation, nous disposons certes d'une structure CP_locale, mais cette structure n'est pas cohérente en général et le graphe qu'elle représente n'est pas planaire. De ce fait, un calcul des intersections entre les arêtes codées dans cette structure s'impose. Suite à ce calcul, de nouveaux brins, sommets et arêtes vont apparaître : chaque point d'intersection entre deux arêtes se traduira en général par l'insertion dans la structure de données locale d'un nouveau sommet, de quatre brins et de deux arêtes (cf. figure 5).

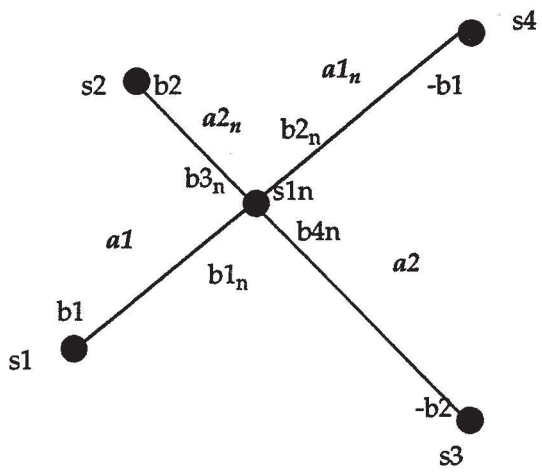


Figure 5 : Mises à jour locales lorsque deux arêtes s'intersectent

L'insertion de ces entités ne produira de structure CP_locale cohérente que si l'insertion des sommets et des brins se fait en respectant le x-y-ordre des sommets et le σ-ordre des brins autour d'un même sommet.

Pour trouver les points d'intersection entre N arêtes, la méthode la plus brutale consiste à tester systématiquement l'intersection entre toutes les arêtes prises deux par deux. Une méthode de ce type a une complexité en $O(N^2)$. Nous nous sommes donc intéressé à l'algorithme de Bentley-Ottmann [BENT 79] qui offre de meilleures performances théoriques.

a. L'algorithme de Bentley-Ottmann

L'algorithme de Bentley-Ottmann exploite et gère deux ordres : le x-y-ordre déjà défini et le y-ordre (cf. figure 6). Le y-ordre est défini par rapport à une droite verticale (x=c) appelée droite de balayage. L'ensemble des arêtes est alors scindé en deux sous-ensembles : celui des arêtes dites actives qui coupent la droite de balayage, et celui des arêtes non actives.

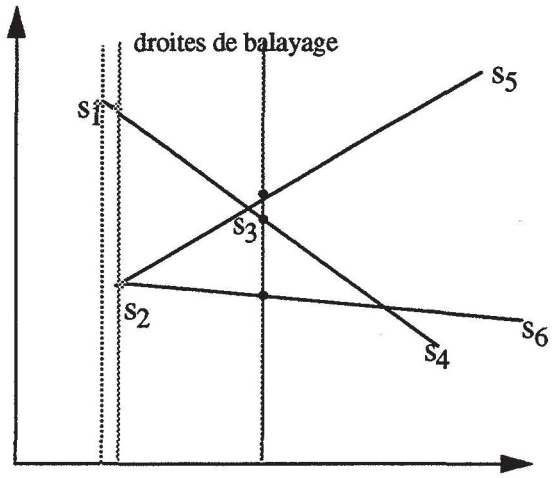


Figure 6 : y-ordre et droites de balayage

Le y-ordre est défini sur l'ensemble des arêtes actives : il classe ces arêtes suivant l'ordonnée croissante de leur point d'intersection avec la droite de balayage.

Cet algorithme utilise une droite de balayage verticale se déplaçant dans le sens des abscisses croissantes. Il se base essentiellement sur deux remarques :

- les seules droites de balayage utiles sont celles qui passent par un sommet ou par un point d'intersection déjà connu. Les modifications du *y-ordre* se font en éliminant les arêtes mortes et en insérant les arêtes naissantes, qui deviennent actives. Rappelons qu'une arête est naissante en son premier sommet et qu'elle meurt en son second sommet (cf. section 1.2 de ce chapitre).

- deux arêtes ne peuvent se couper que si et seulement si elles sont contiguës selon le *y-ordre* pour une droite de balayage donnée.

L'algorithme qui exploite ces deux remarques et la structure CP_locale produite par l'étape d'initialisation est d'abord décrit ci-dessous de façon informelle :

- la droite de balayage est initialement vide, puis elle est déplacée de la gauche vers la droite; elle balaie successivement tous les sommets et tous les points d'intersection, qui sont trouvés et insérés dans le x-y-ordre au fur et à mesure.

- au cours du balayage, le *y-ordre* est généré ainsi : en chaque sommet ou point d'intersection, les arêtes mortes sont enlevées et les arêtes naissantes sont insérées en bonne place. Lors de cette modification, certaines arêtes deviennent contiguës dans le *y-ordre* (cf. figure 7); l'adjacence entre deux arêtes est une condition nécessaire pour qu'elles s'intersectent; il suffit donc pour chaque nouvelle adjacence, de tester les arêtes contiguës.

- si deux arêtes deviennent adjacentes et convergent vers la gauche, alors elles ont peut-être un point commun, mais il est déjà connu. Ces deux arêtes ont déjà été adjacentes dans le y-ordre, il est donc inutile de tester leur éventuelle intersection.

- si deux arêtes deviennent adjacentes, convergent vers la droite, et si leurs sommets de mort ne sont pas identiques, alors leurs droites support se coupent en un point situé à droite par rapport à la droite de balayage. Si ce point appartient bien aux deux arêtes, alors on crée un nouveau sommet et on remplace chacune des deux arêtes par deux nouvelles arêtes. Le point d'intersection devient un sommet comme les autres et il est inséré dans le x-y-ordre.

- la classification des sommets en sommets de naissance (cf. figure 7 a), sommets de vie (cf. figure 7 b) ou sommets de mort (cf. figure 7 c), permet d'étudier les cas où deux arêtes deviennent contiguës pour un sommet S définissant le y-ordre. Pour un sommet S, nous appelons active basse l'arête immédiatement en dessous, et active haute l'arête immédiatement au dessus. Nous distinguons deux cas :

  - si S est un sommet de mort et si l'active basse et l'active haute existent, alors elles deviennent contiguës. Si, de plus, elles convergent à droite, alors leur intersection est testée.

  - si S est un sommet de vie ou de naissance, alors l'arête naissante la plus basse et l'active basse d'une part, l'arête naissante la plus haute et l'active haute d'autre part deviennent contiguës. Quand ces arêtes contiguës convergent à droite, leur intersection est testée.

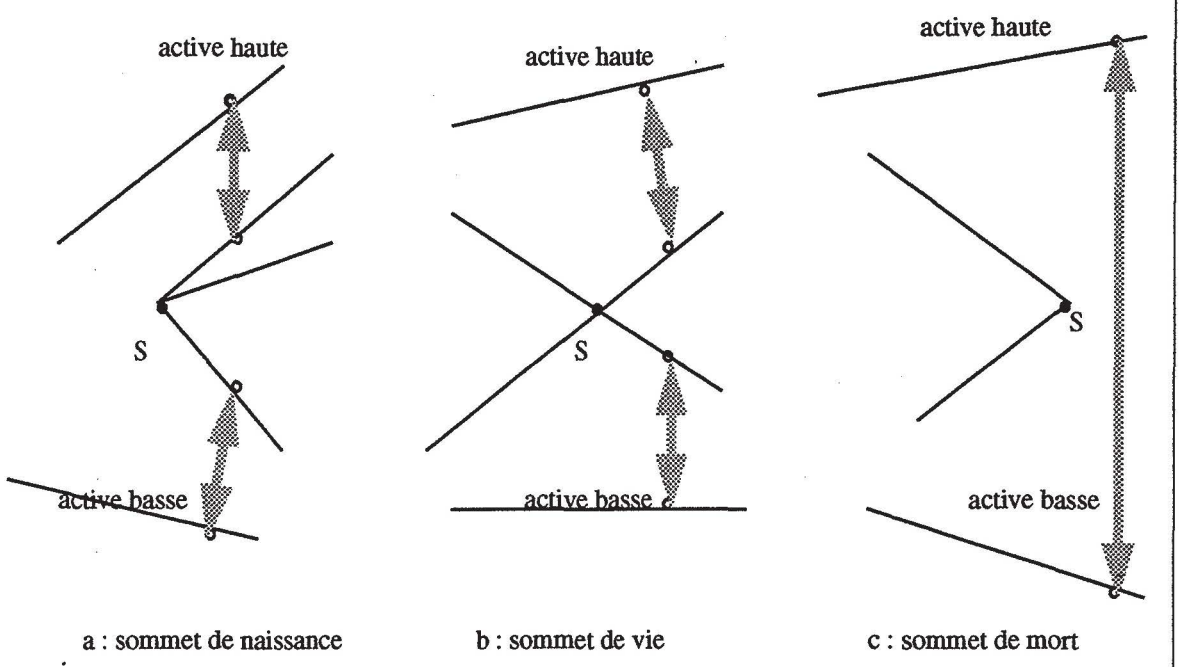


Figure 7 : adjacences nouvelles selon le y-ordre

Nous donnons ci-dessous une description procédurale de l'algorithme de Bentley-Ottmann :

insérer dans le x-y-ordre tous les sommets connus, ordonnés
initialiser le y-ordre comme vide
pour** chaque sommet S, pris selon le x-y-ordre, **faire

- *cas S est un sommet de mort :*
  - *enlever les arêtes mortes du y-ordre*
  - *comparer(active haute, active basse)*
- *cas S est un sommet de naissance :*
  - *insérer les arêtes naissantes dans le y-ordre*
  - *comparer(la plus basse des nouvelles nées, active basse)*
  - *comparer(la plus haute des nouvelles nées, active haute)*
- *cas S est un sommet de vie :*
  - *enlever les arêtes mortes du y-ordre*
  - *insérer les arêtes naissantes dans le y-ordre*
  - *comparer(la plus basse des nouvelles nées, active basse)*
  - *comparer(la plus haute des nouvelles nées, active haute)*

finpour

L'algorithme de la procédure comparer (a, b : arête) est le suivant :

si *(a et b existent) et (a et b convergent vers la droite) et*
(leurs sommets de mort sont distincts) et (a et b se recouvrent en y)
alors
si *(i point d'intersection des droites support de a et b existe)*
alors
mettre à jour les structures de données de carte planaire
insérer i dans le x-y-ordre
finsi
finsi

Le test de convergence de deux arêtes vers la droite utilise deux données attachées aux brins : la donnée *Pente* et la donnée *Cadre*. En fait, la donnée *Cadre* utilise la partition du plan euclidien, rapporté à un repère dont l'origine est le sommet d'incidence du brin, en huit régions : quatre sur les axes et quatre hors des axes. Quand cette donnée, prise aux brins de naissance de deux arêtes a et b, ne suffit pas de décider de leur convergence vers la droite nous utilisons la donnée *Pente* associée aux brins. Le test de recouvrement des deux arêtes utilise les pentes de leurs brins de naissance et leurs sommets de naissance (équations cartésiennes) pour vérifier si oui ou non les deux sommets d'une arête sont de part et d'autre de la droite support de l'autre.

Pour représenter le y-ordre, nous avons créé et utilisé une structure de données appelée *L_actives* caractérisée par un champ de type *Arête* (représentant une arête active) et deux champs de type *L_actives* qui permettent d'accéder à l'active immédiatement inférieure et à l'active immédiatement supérieure. Inversement, chaque structure de type *Arête* contient un champ de type *L_actives* qui permet de la situer dans le y-ordre. Ce jeu de structures de données complété par la mise en place d'une variable globale de type *L_actives*, dont l'adresse est celle de l'enregistrement contenant la plus petite arête active, nous permet d'effectuer efficacement les opérations élémentaires :

- insertion d'une nouvelle arête active,
- extraction d'une arête qui a cessé d'être active,
- recherche d'une arête immédiatement inférieure ou supérieure à une arête donnée, selon le y-ordre,

opérations incontournables dans l'algorithme de Bentley-Ottmann [BENT 79].

Une analyse de la complexité faite par les auteurs de [BENT 79] attribue à l'algorithme de Bentley-Ottmann une complexité en O((N+K)log(N)), où N est le nombre de segments initiaux et K le nombre d'intersections entre arêtes.

b. Cohérence de la CP_locale

En modélisation à l'aide des cartes planaires, le plongement de notions topologiques dans le plan est une source d'incohérences. Certains auteurs choisissent de privilégier la topologie avant tout et construisent les cartes planaires uniquement en utilisant les opérateurs d'Euler. Ces opérateurs sont de nature purement topologique et chacun d'entre eux pris séparément respecte la relation d'Euler-Poincaré, gage de la cohérence des cartes combinatoires.

Or, nous avons choisi les cartes planaires justement pour leur capacité à prendre en compte à la fois la géométrie et la topologie; et l'utilisation des opérateurs d'Euler ne garantit pas la cohérence entre données numériques et données topologiques. Comme nous déduisons la topologie à partir de la géométrie dans nos algorithmes de construction de carte planaire, nous sommes confronté à deux problèmes : le premier est celui de la cohérence de la topologie sous-jacente à la structure CP_locale et le second est celui de la cohérence de la topologie avec les données numériques.

A ce stade de notre réflexion, nous pensons qu'il est important de donner quelques précisions sur la nature des segments géométriques fournis en entrée de l'algorithme de construction des cartes planaires. Contrairement à certaines applications dans le domaine de la modélisation de solides, ces segments ne proviennent pas d'une autre représentation (CSG ou autre). Donc, à la fin de la construction, nous n'avons pas à assurer la cohérence de la carte planaire produite par rapport à un angle de vue (ou de projection) d'un solide tri-dimensionnel par exemple. Dans notre application, les segments fournis en entrée sont saisis par manipulation directe d'un certain nombre d'outils bi-dimensionnels (segment, ligne brisée, rectangle, dessin à main levée). Le tracé initial est pré-traité (cf. section 2.3.3. de la partie C du chapitre III). Le pré-traitement produit des segments recalés suivant des directions angulaires choisies par l'utilisateur.

L'algorithme de Bentley-Ottmann est une source d'incohérences numériques puisque les imprécisions numériques, inhérentes au calcul flottant, sont propagées lors de la progression de la droite de balayage. Michelucci a démontré que l'algorithme de Bentley-Ottmann est assez sensible aux imprécisions numériques : une erreur fatale peut interrompre le déroulement des programmes utilisant la méthode de Bentley-Ottmann, alors que la méthode brutale testant les segments deux à deux arrive toujours à son terme [MICH 87]. Il indique aussi que la résolution des problèmes numériques règle du même coup les problèmes d'incohérences.

Parmi les travaux effectués pour lutter contre les problèmes d'imprécision numérique nous pouvons citer :

- [GREE 86] qui décrit une approche discrète dans laquelle les points d'intersection entre segments, dont les extrémités sont les noeuds d'une grille entière, sont calculés en arithmétique rationnelle par la méthode de Bentley-Ottmann [BENT 87], puis "tirés" sur les noeuds les plus proches de la grille.

- [MORE 90] qui propose une arithmétique exacte-réticente basée sur une arithmétique mixte flottante/rationnelle qui ne fait appel à l'exactitude du calcul rationnel qu'en dessous de certains seuils calculés à l'avance.

- le travail réalisé par une des équipes de notre laboratoire à l'Ecole des Mines de Saint-Etienne qui met en oeuvre une arithmétique rationnelle paresseuse [BENO 93a]. Cette arithmétique retarde les calculs rationnels exacts jusqu'à ce qu'ils deviennent indispensables. Elle repose sur une librairie d'arithmétique rationnelle et aussi sur une bonne maîtrise de l'arithmétique des intervalles.

En opposition par rapport à ces méthodes lourdes et gourmandes en temps de calcul, des solutions partielles ont été proposées parmi lesquelles nous pouvons citer l'approche des epsilons qui consiste à confondre deux sommets dès qu'ils sont distants de moins de epsilon [LAID 86].

Entre ces deux alternatives, nous avons préféré la seconde qui a un coût en temps de calcul compatible avec la nécessité d'une interactivité pour notre application. D'autant que nous sommes assez loin, en nombre d'arêtes, des cas réels non aléatoires, testés par Michelucci (4000 arêtes ou plus), dans lesquels l'algorithme de Bentley-Ottmann n'arrive pas à son terme quand est employée une arithmétique flottante sur 64 bits. Nous signalons qu'une version de Bentley-Ottmann paresseuse a été développée par Michelucci au sein de notre département : elle effectue l'essentiel des calculs en flottant et ne met en route la librairie rationnelle exacte que lors de la rencontre de cas particuliers de segments bien repertoriés. Nous serons peut être amenés à l'utiliser si les erreurs fatales et les incohérences se font de plus en plus fréquentes, et donc génantes vis à vis du futur utilisateur de notre application.

Pour l'instant, nous ne renonçons pas à la version flottante ni au modèle BREP des cartes planaires. Dans la suite de cette thèse, nous supposons que la CP_locale produite par la deuxième étape de la construction de carte planaire est cohérente (heureusement cela se produit dans la majorité des cas traités dans le cadre de notre application). Nous continuerons donc à décrire sur cette base l'ensemble des algorithmes et des améliorations dont l'objet est d'étendre le modèle bi-dimensionnel classique des cartes planaires pour en faire un modèle $2D^{1/2}$ hiérarchique.

4.1.3. Construction de la carte planaire globale CP_globale

Comme nous l'avons dit auparavant, la CP_globale est une explicitation de l'information relative aux faces : les brins qui les constituent et l'arbre de leurs inclusions (cf. section 2.). Cette information est contenue sous forme implicite dans toute CP_locale cohérente.

L'algorithme que nous allons présenter, dans un premier temps de façon informelle, procède en un balayage unique de tous les sommets selon le x-y-ordre, en vue de la création des faces, de leur insertion dans l'arbre d'inclusion et du marquage sur chaque brin de la face qui l'emprunte.

Cet algorithme exploite deux données supplémentaires qui viennent s'ajouter à la CP_locale, le x-y-ordre et le σ-ordre :

- la première est une extension de l'ordre entre deux faces soeurs, ordre défini dans la section 2. de ce chapitre. Cette extension consiste à ne pas comparer les brins de naissance des faces, quand celles-là naissent au même sommet, mais à comparer les brins voisins de leur brin de naissance. Cette subtilité est sans conséquence sur l'ordre entre faces soeurs, mais elle assure que toute face externe est immédiatement inférieure à sa première face fille, et assure l'établissement d'un ordre total sur l'ensemble des faces et pas seulement entre faces soeurs. L'algorithme se propose de faire en sorte que les faces soient créées, selon cet ordre, du plus petit au plus grand.

- la deuxième, est une donnée attachée aux sommets. Elle est utilisée pour la localisation rapide d'une arête ou d'un sommet. Elle est appelée arête de butée ou butée tout simplement : pour un sommet de naissance, c'est la première arête active basse ne naissant pas en ce sommet lors du passage de la droite de balayage; autrement (vie ou mort) c'est la première arête, selon le σ-ordre, mourant en ce sommet (cf. figure 8). Cette donnée est calculée et mise en place lors de la construction de la CP_locale. Nous n'en avons pas parlé dans l'étape précédente puisqu'elle n'est utilisée que pour la construction de la CP_globale.

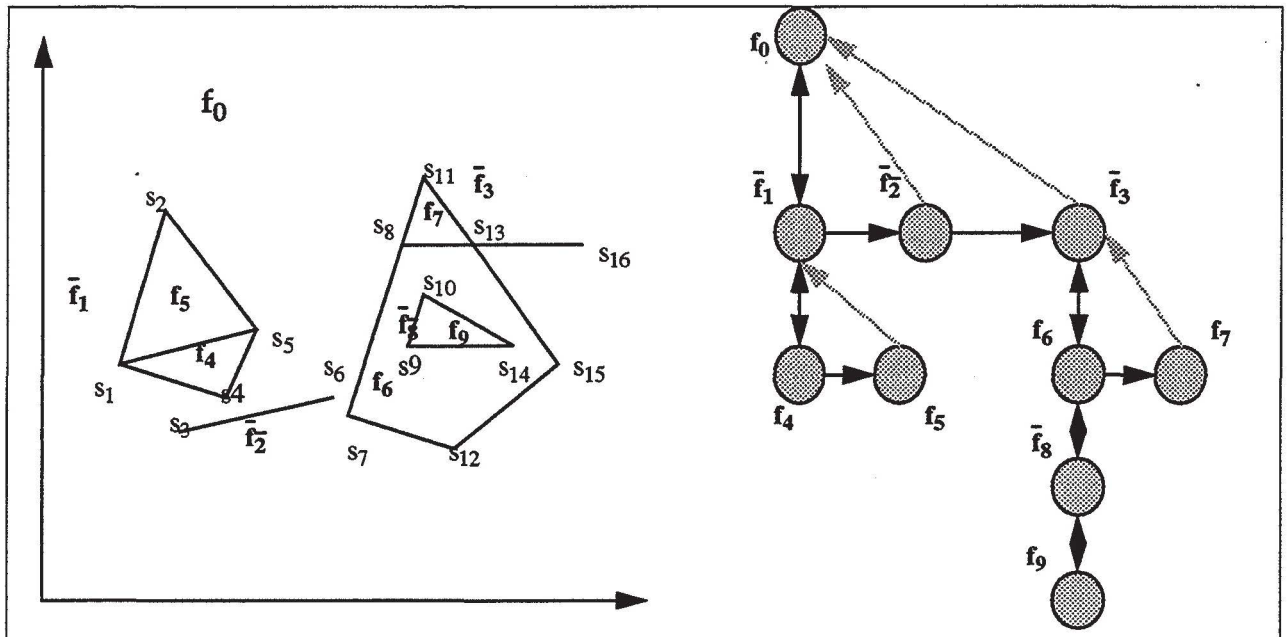


Figure 8 : Butées et arbre d'inclusion des faces

L'algorithme se base également sur quelques remarques importantes :

- les brins appartenant à une face directe sont obtenus par application, au premier brin de la face, des puissances successives de l'application $\sigma^{-1}o\alpha$. De même, il est possible de les obtenir par application, à son dernier brin, des puissances successives de $\alpha o\sigma$ inverse de $\sigma^{-1}o\alpha$,

- le brin de naissance d'une face est toujours le brin numéro un de l'arête qui le porte,

- une face externe ne peut avoir comme brin de naissance qu'un brin partant d'un sommet de naissance.

- une face externe devant être inférieure à toutes les faces internes qu'elle contient, le brin adjacent à son brin de naissance doit être le plus petit selon le σ-ordre, des brins partant de son sommet de naissance,

Voici une présentation informelle de l'algorithme qui permet de construire la CP_globale à partir d'une CP_locale cohérente :

- l'algorithme considère un à un les sommets S, pris selon le x-y-ordre. En chaque sommet, il considère les arêtes A naissantes en ce sommet, prises une à une selon le σ-ordre. Si B2, deuxième brin de A, n'est pas encore marqué, alors nous passerons à l'étape suivante de l'algorithme. Sinon, nous continuerons à considérer les autres arêtes naissantes en S jusqu'à épuisement, auquel cas nous passerons au sommet immédiatement supérieur au sommet courant selon le x-y-ordre.

- quand on arrive à ce point, l'algorithme vient de rencontrer une nouvelle face directe F dont le dernier brin, quand on la parcourt à partir de son brin de naissance, est le deuxième brin B2 de l'arête A. F est alors créée et B2 permet son parcours inverse à l'aide de l'application $\alpha o \sigma$. Pendant le parcours, nous marquons sur chaque brin emprunté son appartenance à F. Le dernier brin marqué est le brin de naissance de la face F : cette information peut donc facilement être intégrée à l'enregistrement correspondant à la face F.

- le type de la nouvelle face F créée peut maintenant être déterminé de la façon suivante : si le sommet S courant est un sommet de naissance et si le brin adjacent au brin de naissance de F (qui correspond au premier brin de l'arête A) est le premier brin dans le σ-ordre autour de S, alors F est externe; F est interne dans les autres cas. Le seul traitement qui reste à faire est celui du positionnement de la nouvelle face créée dans l'arbre d'inclusion des faces. Par définition, la racine de l'arbre d'inclusion des faces contient toutes les faces. Comme un graphe planaire peut être constitué de plusieurs composantes connexes, nous avons choisi comme racine une face virtuelle infinie. Cette face virtuelle est par définition interne. Cette convention nous assure que toute face externe est contenue dans une face interne.

- si F est une face interne, l'algorithme considère la face F2 empruntant le brin deux de l'arête portant le brin de naissance de la face F. F2 existe déjà parce qu'elle a été obligatoirement parcourue avant F. Soit F2 est externe, et alors elle est la parente de F. Soit F2 est interne, et alors F2 est une soeur de F et la parente de F2, connue, est celle de F (cf. figure 8).

• si F est une face externe, l'algorithme considère l'arête de butée A1 du sommet S (cf. figure 9) :

Si A1 n'existe pas, alors F n'est contenue dans aucune face réelle et elle a pour face parente la face interne virtuelle ancêtre de toutes les faces.

Si A1 existe, nous considérons la face F 2 empruntant le brin un de A1. Comme l'arête A1 naît nécessairement avant S, les deux faces empruntant les brins de A1 ont été créées et parcourues avant F.

si F2 est une face interne alors F2 est la face parente de F,

sinon F2 est une face externe et est une soeur de F. Dans ce cas, la face parente de F2, connue, est celle de F.

• la face parente FP de F étant connue, le placement de F en queue de la liste des faces filles de FP peut se faire facilement moyennant une mise à jour aisée du pointeur sur la benjamine des filles de FP.

Cet algorithme est en O(A), A étant le nombre d'arêtes : A tests pour savoir si une arête a été empruntée ou non à sa naissance; 2A marquages de brins; et au plus A créations de faces; le positionnement d'une face dans l'arbre d'inclusion des faces ne nécessite qu'un nombre limité d'opérations.

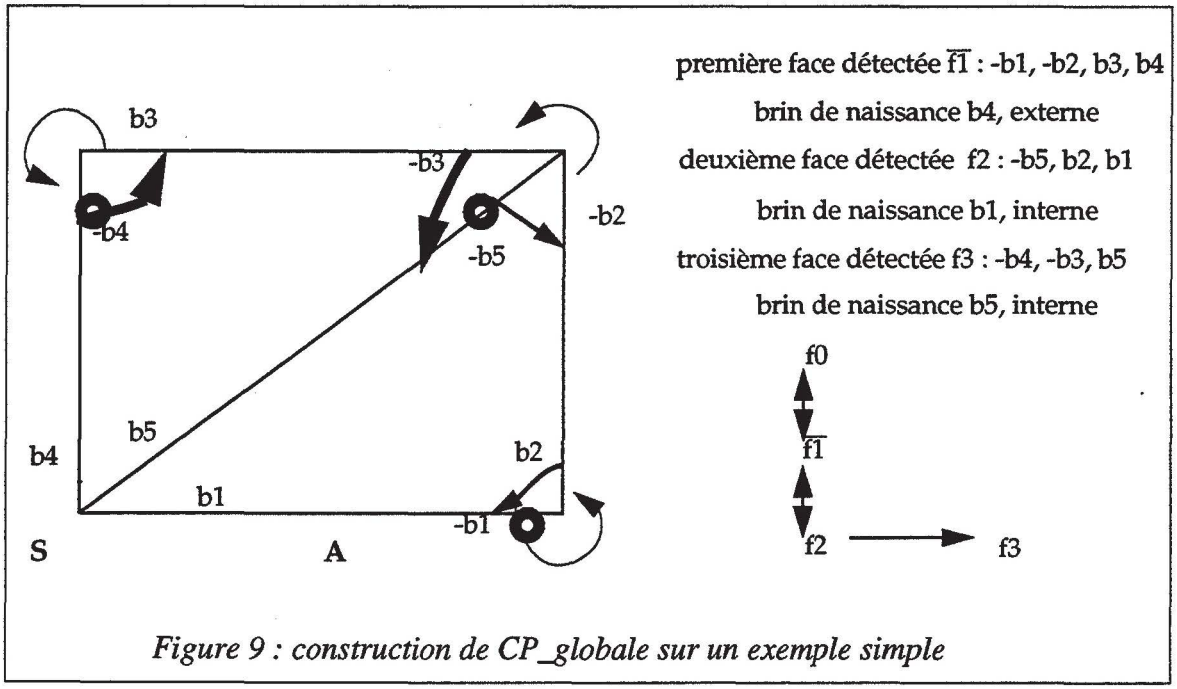


Figure 9 : construction de CP_globale sur un exemple simple

Une description formelle de l'algorithme est donnée ci-dessous :

```
pour chaque sommet S, pris selon le x-y-ordre, faire
    pour chaque arête A naissante en S, prise selon le σ-ordre, faire

    si (la face du brin 2 de l'arête A n'existe pas encore)
    alors
            créer une nouvelle face F
            parcourir les brins de F en les marquant comme empruntés par F
            affecter le brin de naissance de F
            si (S est une naissance) et (A est la première incidente de S)
            alors
                    F est externe
                    si (la butée de S n'existe pas)
                    alors F' := face virtuelle de la carte
                    sinon F' := face du brin 1 de la butée de S
                    si (F' est interne)
                    alors la face parente de F est F'
                    sinon la face parente de F est la face parente de F'
             sinon
                    F est interne
                    F' := face du brin 2 de l'arête de naissance de F
                    si (F' est externe)
                    alors la face parente de F est F'
                    sinon la face parente de F est la face parente de F'
            finsi
            ajouter F à la fin de la liste des filles de la face parente de F
    finsi
```

4.2. La construction incrémentale des cartes planaires

Nous parlerons de construction incrémentale des cartes planaires lorsque l'on dispose d'une carte planaire et qu'on veut rajouter et/ou supprimer un ou plusieurs segments à cette carte. Ce type de construction nous a semblé incontournable surtout dans le cadre de notre application qui est une interface graphique pour une base de connaissances. Ceci pour deux raisons essentielles : le souci de l'interactivité pour la manipulation de l'interface et celui de garder les attributs sémantiques connus de la base de connaissances. Ces attributs sont attachés aux 'objets' qui constituent une carte planaire (arêtes, faces ...).

La construction incrémentale d'une carte planaire consiste en l'insertion et la suppression incrémentale de segments; elle nécessite aussi le calcul dynamique des nouveaux points d'intersection et la mise à jour des structures de données. Comme les opérations d'insertion et de suppression de segments n'induisent que des modifications locales, l'algorithme de Bentley-Ottmann a été adapté pour effectuer une mise à jour efficace de la carte planaire locale. En ce qui concerne la mise à jour de la CP_globale, Ben Amara a étudié et repris dans [BENA 92a] l'algorithme proposé par Gangnet et al. dans [GANG 89]. La solution de [GANG 89] utilise une classification des arêtes ajoutées ou supprimées pour en déduire les modifications à effectuer sur les faces. Ben Amara a montré que cette solution n'était pas adaptée à la sauvegarde des attributs sémantiques attachés aux faces.

Dans ce qui suit, nous présenterons d'abord l'algorithme de mise à jour de la CP_locale. Ensuite, nous présenterons celui de la mise à jour de la CP_globale proposé par Ben Amara.

4.2.1. Mise à jour de la CP_locale

Considérons d'abord le cas de la suppression d'une (ou plusieurs) arête. La suppression des brins de la CP_locale est triviale. Par contre, la mise à jour des butées, utiles lors de la construction de la CP_globale, l'est moins. En effet, certaines arêtes supprimées peuvent être les butées de certains sommets. L'ajout de nouvelles arêtes est par contre moins trivial : non seulement il faut prendre la précaution de mettre à jour les butées correctement, mais il faut aussi déterminer par calcul les nouveaux points d'intersection. Les algorithmes de mise à jour seront présentés uniquement de façon informelle. Nous considérerons successivement le cas de l'ajout d'une arête à une carte planaire et le cas de la suppression d'une arête d'une carte planaire.

• cas de l'insertion

Les étapes associées à l'insertion d'une arête dans une carte planaire existante sont les suivantes (cf. figure 10) :

• les deux sommets, extrémités du segment, sont insérés dans la liste des sommets selon le x-y-ordre.

• la droite de balayage, initialement vide, est déplacée de la gauche vers la droite; elle balaie successivement tous les sommets jusqu'au premier sommet de la nouvelle arête et fait la mise à jour de la pile des arêtes actives. Durant cette phase, il n'existe pas de nouveaux points d'intersection.

• à partir du premier sommet de la nouvelle arête, l'algorithme de Bentley-Ottmann est appliqué pour insérer les nouveaux points d'intersection et ordonner les brins autour des sommets. L'algorithme de Bentley-Ottmann s'arrête quand le dernier sommet de la nouvelle arête est atteint.

- **cas de la suppression**

Considérons maintenant la suppression d'une arête d'une carte existante. Intuitivement l'algorithme de suppression est le suivant :

- supprimer les deux brins de l'arête et faire la mise à jour des sommets concernés.

- déplacer la droite de balayage de la gauche vers la droite; elle balaie successivement tous les sommets et fait la mise à jour des butées. Si un sommet n'a plus de brin, il est alors éliminé.

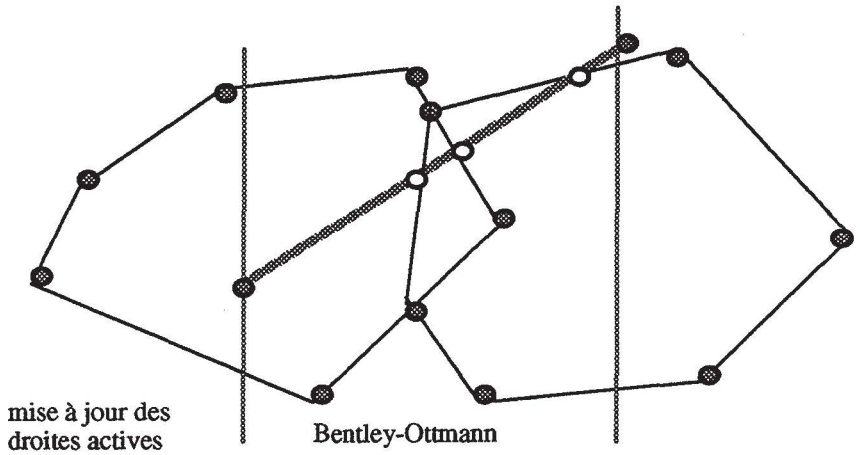


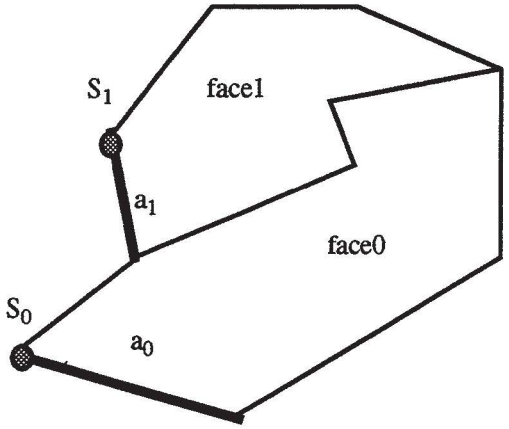
Figure 10 : Mise à jour de la CP_locale suite à l'ajout d'une arête

En cas d'ajouts ou de suppressions multiples de segments, la première étape de chacun des deux algorithmes est effectuée d'un seul bloc sur l'ensemble des segments ajoutés ou supprimés. Pour les deux dernières étapes de l'algorithme d'insertion de plusieurs arêtes, le plus petit sommet de tous les segments ajoutés remplacera le plus petit sommet de la nouvelle arête ajoutée, et le plus grand sommet de tous les segments ajoutés remplacera le plus grand sommet de la nouvelle arête ajoutée. Dans le pire des cas, ces deux algorithmes d'insertion et de suppression d'arêtes présentent une complexité similaire à celle de l'algorithme de Bentley-Ottmann; c'est à dire en O((N+K)log(N)) , où N est le nombre d'arêtes et K le nombre d'intersections entre elles.

4.2.2. Mise à jour de la CP_globale

L'algorithme choisi pour la mise à jour de la CP_globale consiste en une entière reconstruction de l'arbre d'inclusion des faces et en sa comparaison avec l'ancien arbre préalablement sauvegardé (cf. figure 12). Comme nous pouvons le constater d'après ce qui a été exposé dans le paragraphe 4.1.3., l'algorithme de construction de l'arbre d'inclusion des faces est d'une complexité acceptable. Recourir à la solution proposée par Ben Amara, nous permet de mettre à jour l'arbre d'inclusion des faces tout en gardant la sémantique attachée à chacune de ses faces malgré la brutalité de l'approche.

Cette solution se base sur la remarque suivante : toute face de carte planaire est identifiée par son sommet de départ (le plus petit sommet appartenant à la face selon le x-y-ordre) et par son arête de départ (la plus petite arête selon le σ-ordre autour du sommet de départ) (voir la figure 11).



face0 est identifiée par le sommet S_0 et l'arête a_0

face1 est identifiée par le sommet S_1 et l'arête a_1

Figure 11 : identification des faces

Tenant compte de cette remarque, l'algorithme global de mise à jour des structures CP_locale et CP_globale peut être résumé de la manière suivante :

pour toute *modification de la carte planaire* ***faire***

sauvegarder l'ancien arbre d'inclusion
mettre à jour la CP_locale
construire le nouvel arbre d'inclusion
pour chaque *face du nouvel arbre d'inclusion* ***faire***
identifier la face dans l'ancien arbre d'inclusion
si *la face existe* ***alors***
marquer la face dans l'ancien arbre d'inclusion
transférer les attributs associés à cette face de l'ancien arbre vers le nouveau
finsi
finpour
détruire l'ancien arbre d'inclusion

finpour

L'identification de la face dans l'ancien arbre d'inclusion consiste à chercher la face qui a le même sommet de départ et la même arête de départ. Chaque nouvelle face créée possédera la sémantique par défaut associée aux faces de la carte planaire modifiée.

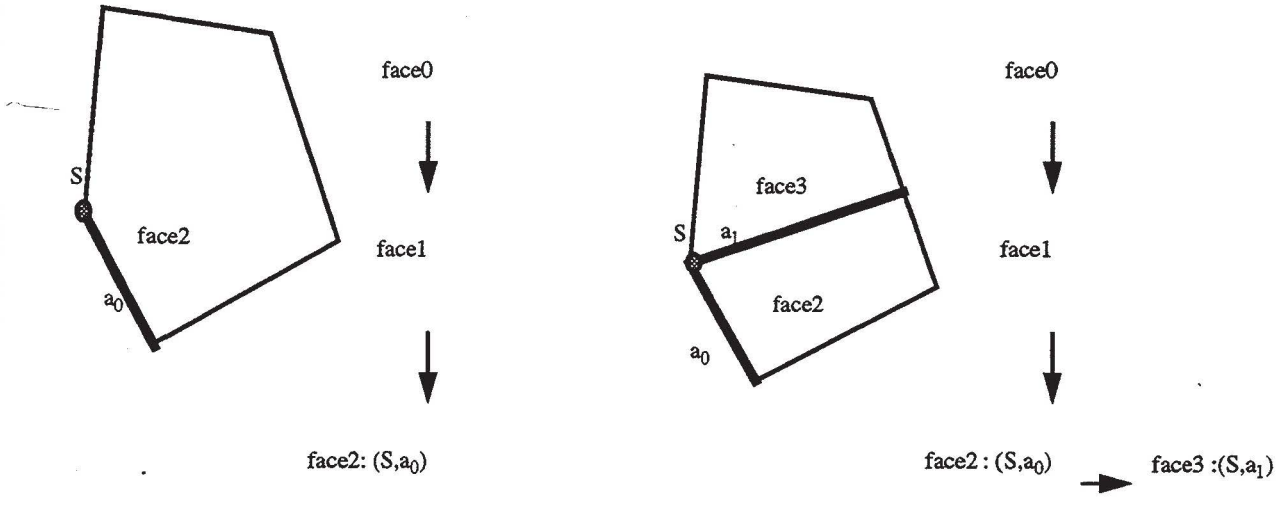


Figure 12 : mise à jour de l'arbre d'inclusion des faces après ajout de l'arête a_1

L'algorithme de construction incrémentale de la carte planaire globale nécessite une reconstruction de l'ensemble des faces (O(N)) et une identification des faces de l'ancienne arborescence dans la nouvelle (O(N)). Cet algorithme a donc une complexité en O(N), où N est le nombre d'arêtes.

5. CONCLUSION

Ce chapitre a présenté des rappels sur les cartes planaires. Plusieurs raisons ont motivé le choix de la structure de données 'carte planaire' pour bâtir la modélisation que nous présenterons dans le chapitre II. D'abord, L'algorithme de construction statique des cartes planaires qui, en plus de son efficacité, construit une structure de données supportant à la fois la topologie et la géométrie et à laquelle il est aisé d'attacher une sémantique. Ensuite, l'algorithme de construction incrémentale rend la notion de carte planaire plus attrayante pour des applications de type interactif comme celle que nous présenterons dans le chapitre III.

Rappelons ici que nous sommes conscients des effets pervers de l'utilisation d'une arithmétique flottante et de l'atteinte qu'elle porte aux propriétés du plan euclidien. Parmi les solutions qui ont déjà été proposées pour ce problème, nous avons opté pour la méthode dite des epsilons, avec un epsilon compatible avec la capacité humaine de discernement entre pixels voisins d'un écran d'ordinateur. Les études récentes sur l'arithmétique rationnelle et l'arithmétique paresseuse garantissent une certaine robustesse et la cohérence de la structure de données 'carte planaire'. Cependant, les temps de réponse qu'elles procurent sont encore loin d'être satisfaisants pour des applications de type interactif, avec les technologies matérielles en notre posséssion.

CHAPITRE II

UN MODÈLE $2D^{1/2}$ HIÉRARCHIQUE BASÉ SUR LES CARTES PLANAIRES

A. INTRODUCTION

La notion de carte planaire doit son succès à sa capacité à modéliser la topologie de scènes assez complexes. Des extensions ont été proposées pour les cartes planaires. On parle alors de cartes généralisées de dimension n ou n-G-cartes [LIEN 88], [ELTE 91]. Ces extensions ont pour but d'offrir un support pour modéliser la topologie de surfaces orientables ou pas, fermées ou pas, de volumes ... Ainsi, un modeleur tri-dimensionnel basé sur la notion de carte généralisée a été développé à Strasbourg [BERT 92].

Classiquement, dans le domaine de la représentation par les frontières, un solide est défini par une subdivision d'une surface fermée orientable. D'un point de vue informel, une subdivision de surface peut être définie par une partition de cette surface en faces, arêtes et sommets [LIEN 88]. Des travaux ont été effectués pour expliciter un certain nombre d'opérateurs permettant de construire des partitions de solides valides. Parmi ces opérateurs, les plus connus sont les opérateurs d'Euler [MANT 82]. L'objectif de tels opérateurs est de garantir que les constructions faites par application de ces opérateurs représentent bien des solides. Inversement, Mantyla et Sulonen proposent dans [MANT 82] un algorithme d'inversion permettant de décomposer tout solide valide (dans un certain sens) en une suite de constructeurs d'Euler. En raison de l'aspect très élémentaire de ces opérateurs (construire une arête à partir de deux sommets, découper une arête en deux et créer un nouveau sommet ...), leur utilisation devient fastidieuse dès que le solide à modéliser est assez complexe. D'où, l'apparition du concept de macro-opérateurs dits Eulériens puisqu'ils s'expriment en fonction des opérateurs d'Euler de base [ANSA 85]. Le concept de macro-opérateur apporte une définition compacte et efficace pour la mise à jour d'objets solides.

Des modeleurs solides utilisent les cartes planaires en raison de certaines nécessités de visualisation (remplissage de faces ou régions, production de vues ...) et de calcul de caractéristiques géométriques ou physiques (volume, masse, centre de gravité ...) [MICH 87], [BENO 93] ... Souvent, les cartes planaires sont obtenues à partir de descriptions d'arbre de construction CSG. En effet, un grand nombre de modeleurs tri-dimensionnels combinent les deux approches CSG et représentation par les frontièes (BREP) tout en privilégiant l'une des deux par rapport à l'autre. Un des modeleurs privilégiant l'approche BREP est GWB (Geometric WorkBench) [MANT 82].

Parmi les applications basées sur le concept de carte planaire, on peut citer un logiciel permettant la saisie d'esquisses d'architecture [MICH 84]. Moissinac a utilisé les cartes planaires pour des logiciels d'animation [MOIS 84]. Un autre exemple d'application est la saisie d'objets bi-dimensionnels avec l'instauration d'un ordre entre eux (derrière, devant ...) [BRAQ 88], [BRAQ 91] ... Cette application permet de définir des scènes $2D^{1/2}$ à l'aide d'une bonne gestion de cet ordre. Baudelaire et Gangnet montrent dans [BAUD 89] comment un utilisateur peut définir un nouvel objet bi-dimensionnel moyennant des opérations de coloriage, de masquage, de collage ... sur des objets d'une scène $2D^{1/2}$. Arquès et Janey présentent dans [ARQU 92] une modélisation de cartes planaires permettant la synthèse d'images d'un relief montagneux à partir de l'arborescence formée par son réseau fluvial.

Dans ce qui suit, nous allons présenter dans la partie B un modèle $2D^{1/2}$ basé sur les cartes planaires. Nous présenterons ensuite l'aspect hiérarchique de notre modèle dans la partie C. Le modèle $2D^{1/2}$ hiérarchique auquel nous consacrons ce chapitre utilise l'ensemble des algorithmes que nous avons exposés dans le premier chapitre.

B. VERS UN MODÈLE $2D^{1/2}$ BASÉ SUR LES CARTES PLANAIRES

1. INTRODUCTION

Nous désignons par l'appellation "modélisation $2D^{1/2}$" toute modélisation plus riche qu'une modélisation bi-dimensionnelle mais qui n'a pas la puissance de description d'une vraie modélisation tri-dimensionnelle. Un exemple de modèle $2D^{1/2}$, utilisé dans le domaine des cartes planaires, est celui présenté dans [DOME 91]. Ce modèle définit des objets bi-dimensionnels tout en gérant un ordre établi entre eux : objet 1 est situé entre objet 2 et objet 3 ... Cet ordre permet, notamment, de gérer la visualisation des objets bi-dimensionnels ainsi décrits : objet 2 est complètement masqué, objet 1 et objet 3 se chevauchent et une partie de objet 1 est cachée ...

Dans un bon nombre de domaines d'application (architecture, génie civil ...) des représentations planes suffisent pour comprendre le travail qui doit être réalisé. Cependant, des facteurs tels que la complexité d'un immeuble ou la superposition d'informations (murs, circuits électriques, canalisations d'eau ...) dans de telles représentations planes rendent illisible leur présentation sur un écran d'ordinateur. De ce fait, notre modèle $2D^{1/2}$ se propose d'apporter une solution de modélisation et de visualisation pour ce genre de scènes sans être pour autant un vrai modèle tri-dimensionnel.

Le modèle $2D^{1/2}$ auquel nous consacrons cette partie gère la superposition d'objets regroupés en cartes planaires indépendantes caractérisées, chacune, par une sémantique et par un niveau. Le mot niveau a le sens qu'on lui attribue dans un plan d'architecte.

Pour garder le sens global des scènes modélisées, le modèle doit assurer des liens de type sémantique entre les objets répartis sur les différentes cartes planaires (une ligne électrique et une canalisation d'eau se trouvent dans la même pièce, le circuit de chauffage central d'un étage est aligné verticalement avec celui de l'étage en dessous ...).

Nous avons distingué deux aspects dans notre modélisation $2D^{1/2}$:

- le premier est induit par des relations entre objets d'un même étage qui ne font pas partie de la même classe sémantique (la classe sémantique d'un bâtiment, la classe sémantique des canalisations d'eau, la classe sémantique des installations électriques ...),
- le deuxième est induit par des relations entre objets d'étages différents, qui, donc, appartiennent à des plans strictement parallèles.

Ces deux aspects posent des problèmes de manipulation et de visualisation (dus à la superposition d'objets lors de leur représentation dans le plan) et des problèmes de mise en place et de gestion des relations dont nous venons de parler.

Pour le premier aspect de notre modélisation, nous proposons une solution qui consiste en une décomposition, selon la sémantique, des données planes en plusieurs cartes planaires superposables. Pour le second aspect, nous offrons une solution intitulée représentation tri-dimensionnelle, qui n'a rien cependant d'une vraie modélisation tri-dimensionnelle. Dans ce qui suit, nous présenterons successivement ces deux solutions.

2. DÉCOMPOSITION D'UN NIVEAU PLAN EN CARTES PLANAIRES

La structure de données 'carte planaire' présentée dans le chapitre I est construite à partir d'un lot de segments. Pour une application donnée, chacun des segments fourni en entrée des algorithmes de construction de carte planaire possède un sens sémantique : mur, câble électrique ... L'idée de base de la décomposition consiste à partitionner les segments de départ d'un niveau plan en lots de segments caractérisés, chacun, par une classe sémantique particulière et à lancer, séparément, les algorithmes de construction des cartes planaires sur chacun des lots. Cela produit des cartes planaires indépendantes que nous superposons (cf. figure 13).

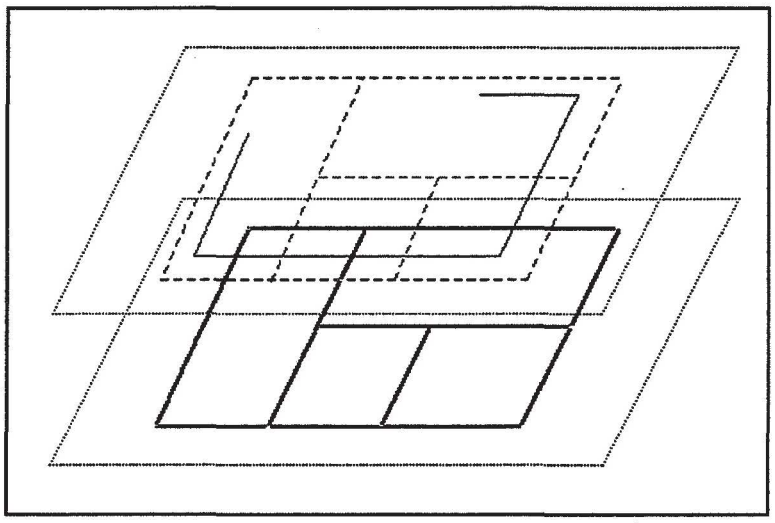


Figure 13 : superposition de cartes planaires associées à des sémantiques différentes

2.1. Décomposition et sémantique - structures de données et algorithmes

Si l'on considère une application telle que les objets manipulés appartiennent à **k** sémantiques différentes, la décomposition d'un niveau plan selon la sémantique nous conduit à représenter un niveau plan par une structure du type : **tableau TCP de k cartes planaires.**

Ainsi, en construction statique, tous les segments possédant la même sémantique sont traités dans un même lot et stockés dans le même élément du tableau TCP. De même, en construction incrémentale, un segment ajouté ou supprimé est, au regard de sa sémantique, ajouté ou supprimé à l'un des k éléments du tableau TCP.

Pour maintenir de façon aisée la cohérence entre les différentes cartes planaires indépendantes associées à un même niveau plan, nous privilégions une sémantique par rapport aux autres et la carte planaire qui la représente par rapport aux autres cartes planaires. Nous appelons cette carte planaire privilégiée la carte planaire de base. Elle est, en quelque sorte, la propriétaire de l'ensemble des objets sémantiques. Par exemple, dans un plan d'architecte, la carte planaire de base sera celle représentant le gros oeuvre (murs, pièces, ouvertures ...).

De plus, l'indépendance de ces cartes planaires est compensée par la mise en place de liens sémantiques entre les objets qu'elles modélisent. Ceci permet de rétablir le sens global de la scène en situant chaque objet sémantique la composant par rapport à ceux choisis comme repères dans la carte planaire de base. Notre solution concernant ces liens consiste à les matérialiser, au niveau de chaque objet, par un champ *Liens* pointant vers le premier enregistrement de ses liens. Chaque structure *Liens* contient un tableau à deux éléments de type *Connexion* et possède un champ nommé *type* qui code le type sémantique de la relation entre les deux objets auquels les structures *Connexion* permettent d'accéder (cf. figure 14). Le type sémantique de la relation peut être : se trouver à l'intérieur de, longer, existence d'un lien technique spécial ...

La structure de données *Connexion* que nous avons utilisée est la suivante :

```
struct Connexion
        {
                struct objet *obj;
                int indice;
                struct liens *lien;
        }

struct Liens
        {
                int type;
                struct connexion tab_conn[2];
        }
```

Un objet de la scène est caractérisé par sa sémantique cp, $0 \leq cp \leq k-1$ et par son identificateur. Un champ de la structure *Objet* permet d'accéder à sa représentation graphique qui peut être une arête, une face, une icône ... De plus, nous connaissons pour chaque objet la structure **liens** de toutes ses connexions et un indice **cote** (0 ou 1) qui nous permet d'initialiser le parcours de ses liens. Les objets sémantiques d'une carte planaire sont représentés par la liste chainée *Objet* qui suit :

```
struct objet
    {
        int cp;
        int identificateur;
        int cote;
        union gr_rep
            {
            struct arete *ar;
            struct face *f;
            struct icone *im;
            ...
            };
        struct liens *liens;
        ...
        struct objet *suivant;
    }
```

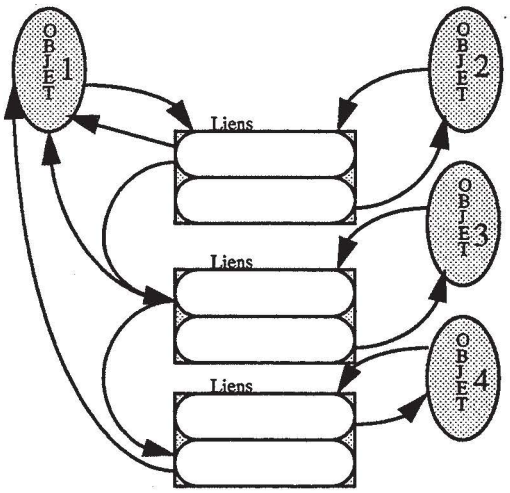


Figure 14 : Les connexions entre objets vues par l'objet 'object 1'

Soit obj un objet ayant des liens sémantiques dont l'accès est initialisé par les données con = obj->liens et cote1 = obj->cote. Un objet obj2 connecté à obj est fourni par la connection courante con et par l'affectation : *obj2 := con->tab_conn[1-cote1].obj.* Le type sémantique de la relation qui lie obj à obj2 est fourni par *con->type.* Le passage de la structure con courante à la suivante parmi celles de obj se fait par le traitement : si cote2 est une variable intermédiaire entière alors :

cote2 := cote1; cote1 := con->tab_conn[cote1].indice; con:= con->tab_conn[cote2].lien.

Des procédures élémentaires opérant sur les structures *Connexion* et *Liens* ont été développées. Elles exploitent les connaissances rapportées ci-dessus. Outre ces procédures, le modèle $2D^{1/2}$ utilise bien sûr les algorithmes de construction statique et incrémentale des cartes planaires présentés dans le chapitre I.

2.2. Les avantages de la décomposition selon la sémantique des données d'un niveau plan

De la décomposition d'un niveau plan selon la sémantique résultent plusieurs cartes planaires indépendantes, chacune d'entre elles représentant une sémantique particulière. La ventilation des segments, selon la sémantique, sur plusieurs cartes planaires offre les avantages que nous détaillons ci-dessous.

L'indépendance de ces cartes planaires a un effet bénéfique sur la cohérence des structures : l'ensemble des arêtes structurées par le concept de carte planaire est stocké dans un élément du tableau TCP et se limite aux seules arêtes ayant la même sémantique. Ce gain potentiel de cohérence vient du fait que l'algorithme de Bentley-Ottmann, source des incohérences numériques, n'est pas lancé sur l'ensemble des segments initiaux. L'algorithme de Bentley-Ottmann est, en effet, lancé indépendamment sur chaque ensemble de segments initiaux de même sémantique et par conséquent il traite un nombre de segments moins important.

Un autre avantage de cette indépendance réside dans le fait que, dès la conception de l'application informatique, une texture particulière peut être affectée au dessin des arêtes de même sémantique. Ceci permet de faire une distinction visuelle entre un nombre important d'arêtes et de faciliter la lisibilité des plans superposés. Pour aboutir à des vues de plans assez lisibles, l'utilisateur peut choisir de visualiser uniquement les objets d'une même sémantique ou de superposer les plans de deux ou plusieurs sémantiques différentes ... Le problème de la désignation des objets étant résolu par indication de la sémantique de l'objet à sélectionner et par désignation directe sur l'écran de l'endroit où il est affiché.

Enfin, le modèle $2D^{1/2}$ basé sur la décomposition des données planes en cartes planaires superposables permet de concevoir des applications prenant en compte plusieurs classes sémantiques. Cette décomposition ne nuit pas au sens global des scènes modélisées grâce aux mécanismes prévus pour l'établissement et la mise à jour de liens sémantiques entre objets des différentes cartes planaires.

3. LA REPRÉSENTATION TRI-DIMENSIONNELLE DE CARTES PLANAIRES

Comme nous l'avons signalé dans la section B.1, une des raisons d'être de la représentation tri-dimensionnelle est d'offrir les moyens permettant :

- la manipulation et la visualisation de données dont les représentations sur un plan se superposent. La représentation tri-dimensionnelle concerne spécialement les données qui appartiennent à des plans strictement parallèles,

- la mise en place de relations entre des objets appartenant à des niveaux plans strictement parallèles. Ces relations pouvant être de type sémantique (existence d'un lien technique particulier ...) ou géométrique (alignement vertical entre deux objets appartenant à des plans différents).

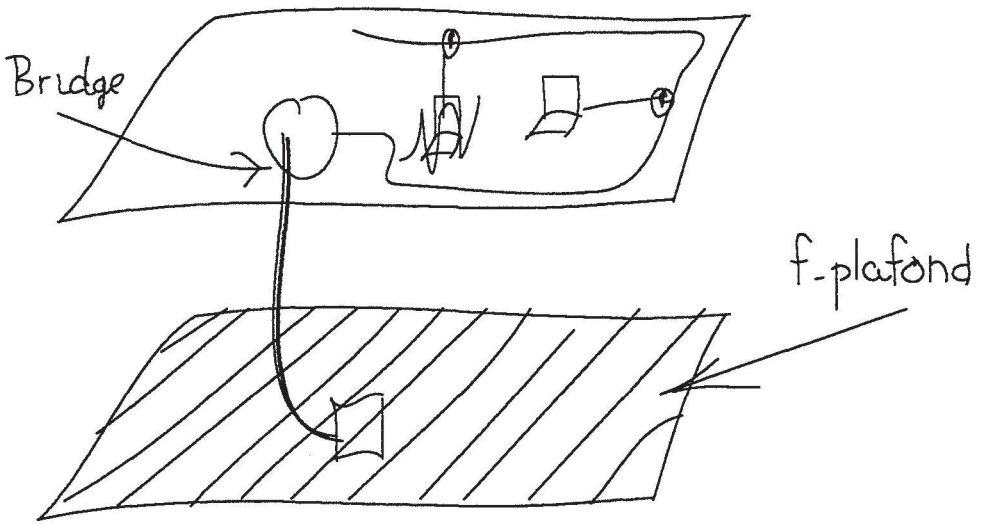


Figure 15 : dessin sur papier d'experts humains

Une solution évidente au problème évoqué dans le premier point aurait été d'opter pour une vraie modélisation tri-dimensionnelle. Sachant que, dans un bon nombre de domaines d'application, des représentations planes suffisent pour comprendre n'importe quelle réalisation, nous avons choisi de garder le modèle bi-dimensionnel décrit dans la partie B de ce chapitre, et de l'étendre en le munissant de moyens de représentation tri-dimensionnels. Ce choix est conforté par le rapport [DARS 90] établi suite à une étude conduite auprès d'experts dans le domaine de la conception des réseaux informatiques (cf. figure 15).

Notre modèle permet ainsi d'exploiter pleinement les possibilités du modèle bi-dimensionnel, et ce de façon classique, pour décrire chacun des plans strictement parallèles

composant une scène. En ce qui concerne la visualisation tri-dimensionnelle de l'ensemble des niveaux plans, leur manipulation et la mise en place de relations entre les objets qui les composent, nous utilisons une transformation dite transformation tri-dimensionnelle et la structure de données *Connexion* décrite dans la section B.2.1.

Dans ce qui suit, nous présenterons d'abord la transformation tri-dimensionnelle. Ensuite, nous décrirons comment est gérée la dualité entre les deux types de vue : bi-dimensionnelle et "tri-dimensionnelle" ou plutôt $2D^{1/2}$ produite par utilisation de la transformation tri-dimensionnelle. Nous terminerons par une description de la manière dont sont prises en compte les relations entre objets de niveaux différents.

3.1. La transformation tri-dimensionnelle

La transformation que nous recherchions devait avoir comme principale caractéristique la simulation de l'aspect tri-dimensionnel en transformant un rectangle quelconque défini par ses deux sommets opposés A et M en un parallélogramme défini par ses sommets opposés B et N (cf. figure 16).

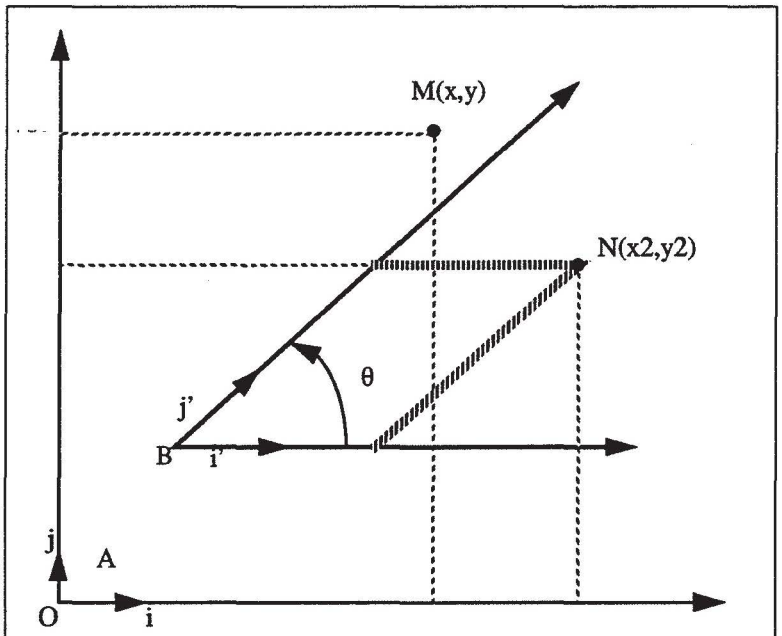


Figure 16 : la transformation tri-dimensionnelle

Nousnoussommesdoncinspirédestechniquesdedessinenperspectivecavalièreetavonschoisi comme transformation tri-dimensionnelle une application affine définie par deux points A et B, un coefficient α dit de réduction, et un angle θ simulant l'aspect tri-dimensionnel. Etant donné un repère R=(O, i, j), cette application affine f est telle que :

(1) $f(A) = B$
(2) $f(i) = i' = \alpha . i$ avec $\alpha > 0$
(3) $f(j) = j' = (\alpha/2) . ((\cos \theta) . i + (\sin \theta) . j)$

Par commodité, nous avons choisi l'angle θ dont le cosinus vaut 0.6 et le sinus vaut 0.8 ($\theta = 53$ degrés à peu près). Par contre, le coefficient α demande plus d'investigation en vue de représenter toute l'information bi-dimensionnelle dans la zone d'écran allouée au dessin "tri-dimensionnel" et de faire en sorte que le dessin "tri-dimensionnel" occupe le maximum possible de cette zone (constater l'étalement vers la droite du parallélogramme "BN" dans la figure 16).

De ce fait, pour un meilleur affichage, le coefficient α doit être recalculé à chaque modification du type insertion/suppression de segment qui est susceptible de modifier la boîte rectangulaire qui englobe la projection, sur l'un des plans, de toutes les données contenues dans les plans strictement parallèles. Un redimensionnement de la zone dédiée à l'affichage "tri-dimensionnel" des données entraîne également un recalcul de α. Or, des besoins algorithmiques nous ont amené à maintenir à jour une donnée supplémentaire attachée à la structure carte planaire : c'est une donnée attachée à chaque face indiquant la boîte rectangulaire minimale, parallèle aux axes, qui l'englobe. Comme les données bi-dimensionnelles sont structurées en cartes planaires, nous avons utilisé avec profit les boîtes englobantes des faces externes, filles directes de la face interne virtuelle, pour calculer efficacement la boîte englobant les dessins de chaque niveau plan. Ainsi, le calcul de α revient à résoudre les deux équations :

$$x2 = \alpha (x - a) + (\alpha/2) (y - b) \cos\theta + c \qquad (I)$$

$$y2 = (\alpha/2) (y - b) \sin\theta + d \qquad (II)$$

avec :

- $\cos\theta = 0.6$ et $\sin\theta = 0.8$,

- (x2, y2) coordonnées dans (O, i, j) du sommet haut droit de la zone IM_RECT allouée au dessin "tri-dimensionnel" de chaque niveau plan,

- (c, d) coordonnées dans (O, i, j) du sommet bas gauche de la zone IM_RECT,

- (a, b) coordonnées dans (O, i, j) du sommet bas gauche de G_ENGLOB boîte rectangulaire englobant toutes les données (carte planaire) contenues dans les plans strictement parallèles,

- (x, y) coordonnées dans (O, i, j) du sommet haut droit de G_ENGLOB,

Si $\alpha 1$ et $\alpha 2$ sont les solutions respectives de (I) et (II), le coefficient α garantissant que tous les dessins tri-dimensionnels rentrent dans l'espace qui leur est alloué, tout en occupant le maximum possible de cet espace, est :

$$\alpha = \min(\alpha 1, \alpha 2)$$

En fait, le coefficient α dépend uniquement de (x - a) et (y - b) qui représentent respectivement la largeur et la hauteur de G_ENGLOB, et de (x2 - c) et (y2 - d) qui représentent respectivement la largeur et la hauteur de IM_RECT. Ce coefficient ne dépend nullement des positions respectives des deux rectangles G_ENGLOB et IM_RECT dans le plan.

3.2. La dualité entre le modèle bi-dimensionnel des cartes planaires et la représentation tri-dimensionnelle - structures de données

Un élément important dans la dualité entre le modèle bi-dimensionnel des cartes planaires et la représentation tri-dimensionnelle est la mise en place d'une fenêtre spéciale dédiée à l'affichage tri-dimensionnel (F3D). Ceci permet de visualiser sous forme tri-dimensionnelle l'ensemble des plans strictement parallèles composant une scène sur F3D tout en ayant une vue bi-dimensionnelle du niveau plan en cours de finalisation sur la fenêtre F2D dite bi-dimensionnelle. Chacun des niveaux plans strictement parallèles pouvant être décrit séparément sur F2D à l'aide du modèle bi-dimensionnel.

L'autre élément de cette dualité se base sur la non-duplication des données : toutes les données sont bi-dimensionnelles et structurées à l'aide du concept de carte planaire. Nous avons vu dans la section B.2.1. comment une structure du type :

```
struct TCP
        {
        struct cplanaire *tcarte[k];
        }
```

pouvait modéliser, à l'aide de cartes planaires, des données planes ayant k sémantiques différentes et appartenant au même niveau plan. La modélisation de données réparties sur plusieurs niveaux plans strictement parallèles peut se faire à l'aide de la structure :

```
struct Liste_carte
        {
                struct cplanaire *tcarte[k];
                struct Liste_carte *plan_precedent;
                struct Liste_carte *plan_suivant;
                int numero_plan;
        }
```

Nous avons dû rajouter un champ supplémentaire à la structure représentant chacun des objets sémantiques de la scène. Ce champ indique le niveau plan auquel appartient l'objet sémantique. Un des avantages que nous en retirons est par exemple une information rapide du type : quand un objet est désigné depuis la fenêtre F3D, savoir s'il figure sur F2D et s'il y a lieu d'opérer un traitement de vidéo inverse sur F2D également. La structure représentant la liste chaînée des objets sémantiques modélisés par une carte planaire caractérisée par sa sémantique et son niveau devient la suivante :

```
struct objet
        {
            struct Liste_carte *lcarte;
            int cp;
            int identificateur;
            int cote;
            union gr_rep
                {
                struct arete *ar;
                struct face *f;
                struct icone *im;
                ...
                };
            struct liens *liens;
            ...
            struct objet *suivant;
        }
```

Pour la visualisation de l'ensemble des niveaux plans, nous avons choisi de ne pas surcharger la fenêtre dédiée à l'affichage tri-dimensionnel, tout en gardant les notions "au dessus de" et "en dessous de". Notre choix s'est porté sur la visualisation simultanée d'au plus trois niveaux plans strictement parallèles et sur la mise en place d'un ascenseur permettant l'accès aux autres niveaux non affichés. Les trois niveaux affichés sont choisis de telle sorte qu'ils constituent le meilleur sous-ensemble de tous les niveaux strictement parallèles encadrant le niveau affiché dans la fenêtre F2D.

La fenêtre F3D, dite tri-dimensionnelle, recueillant le tracé tri-dimensionnel de tous les niveaux plans, a donc été partagée en trois zones rectangulaires ne se chevauchant pas. Chacune d'entre elles doit recueillir le tracé tri-dimensionnel d'un niveau plan. Les dimensions de ces zones sont exactement connues en pixels à partir de la connaissance du dimensionnement courant de F3D et de quelques règles de présentation (marges de droite et de gauche, marges verticales entre les zones réservées à chaque niveau ...).

La tenue à jour d'une variable globale pointant directement sur le niveau affiché dans la plus basse des zones tri-dimensionnelles permet de savoir pour chacun des niveaux plans

affichés dans F3D la transformation tri-dimensionnelle associée. Cette variable est affectée en fonction du niveau affiché de façon bi-dimensionnelle sur F2D. En général, des niveaux différents existent de part et d'autre du niveau affiché dans F2D. La variable globale du tracé tri-dimensionnel est alors affectée de telle sorte que le niveau affiché sur F2D occupe la zone centrale de F3D. De plus, la transformation tri-dimensionnelle générale définie par les équations (I) et (II) de la section B.3.1 est inversible. Les données saisies, depuis F2D ou F3D, sont donc correctement prises en compte et stockées dans les cartes planaires décrivant les niveaux affichés dans F3D, et simultanément visualisées sur F3D et F2D.

3.3. Prise en compte de fonctionnalités inter-niveaux

Sachant que la représentation tri-dimensionnelle nous permet de visualiser et de manipuler plus d'un niveau plan, des fonctionnalités inter-niveaux peuvent être facilement mises en place. En particulier, des relations liant des objets appartenant à des niveaux plans différents peuvent être créées moyennant la sélection des objets qu'elles impliquent et le choix d'une fonctionnalité particulière dans un menu par exemple. Ces relations peuvent être de nature sémantique (existence d'un lien technique comme une gaine technique verticale) ou géométrique (alignement vertical entre deux éléments appartenant à des niveaux situés l'un au dessus de l'autre). Ce genre de relations (connexions) a été implémenté avec succès à l'aide de la structure de données représentée figure 14.

C. INTRODUCTION DE LA HIÉRARCHIE DANS LE MODÈLE $2D^{1/2}$ BASÉ SUR LES CARTES PLANAIRES

1. INTRODUCTION

Dans le domaine de l'informatique graphique, plusieurs travaux ont été menés dans le but de définir des hiérarchies permettant de prendre en compte et de gérer les niveaux de détail. [HOPP 93] fournit un algorithme d'optimisation de maillages d'objets, via la minimisation d'une fonction Energie. Les maillages optimisés ont donc un nombre réduit de sommets dont la connectivité peut changer par rapport à celle de départ. Cet algorithme peut être utilisé soit pour la reconstruction de surfaces à partir de points non structurés, soit pour la simplification de maillages. [GREE 93] propose une méthode de visualisation de scènes complexes par la combinaison de deux types hiérarchiques de structures de données : le tampon de profondeur pour l'espace image et l'arbre octal pour l'espace des objets. Cette méthode a été utilisée pour produire des vues d'un environnement de bureau représenté par 538 millions de polygones. [DeFL 93] présente un modèle combinant les avantages des techniques de triangulation de Delaunay [PREP 88] et ceux de l'organisation hiérarchique des données : détailler un triangle consiste à rajouter des points en son intérieur et/ou sur ses côtés et à calculer la triangulation de Delaunay pour ces points là. Ce modèle a été utilisé pour concevoir et développer une base de données topographique hiérarchique.

Un système complet de synthèse d'images, baptisé ILLUMINES, a été développé dans notre laboratoire [BEIG 89]. Ce système comprend un modeleur de type CSG, plusieurs algorithmes d'élimination des parties cachées et plusieurs processus de rendu . Le modeleur utilise la programmation orientée objet. Il offre la possibilité de définir une hiérarchie parmi les classes d'objets pour gérer des niveaux de détail différents associés à différents positionnements d'un observateur par rapport à la scène. [JAHA 91] expose la méthodologie utilisée pour définir une hiérarchie de classes flexible et évolutive. [BEIG 91] est consacré aux algorithmes et techniques permettant à ce modeleur de produire de façon automatique des vues moins détaillées associées à une agrégation d'objets polygonaux texturés décrits, chacun, de façon détaillée par l'utilisateur. Néanmoins, le problème de la production automatique de vues moins détaillées de scènes complexes reste un des problèmes ouverts de l'informatique graphique.

Parmi les modèles basés sur les cartes combinatoires ayant une structure hiérarchique figure celui proposé dans [ANSA 85]. Ce modèle utilise la face comme moyen de hiérarchisation et permet une description hiérarchique et structurée des solides selon le processus de leur conception. L'utilisateur décrit l'aspect général de son solide par l'ensemble des faces qui le

bordent. Cela produit un graphe d'adjacence de faces FAG_0. Ensuite, les détails peuvent être spécifiés en tant qu'attributs des faces de FAG_0 : FAG_0 se trouve alors expansé en plusieurs sous-graphes FAG_i. Ce processus de description peut être réitéré plusieurs fois.

Dans la littérature, il n'existe pas à notre connaissance de modèle hiérarchique permettant de détailler une partie d'un plan par une carte planaire. L'objectif que nous avons fixé à notre modèle hiérarchique est de permettre la définition de raffinements (détails) sur un ensemble de cartes planaires superposées associées aux différentes sémantiques. Comme nous privilégions la carte planaire de base par rapport aux autres, le passage du niveau de raffinement i au niveau supérieur i+1consiste à extraire de la carte planaire de base la face (ou les faces) à détailler (cf. figure 17). Du fait de la superposition des cartes planaires sémantiques, des objets qui sont la propriété (non exclusive) d'une face de base détaillée peuvent en sortir. Le maintien de la cohérence entre la représentation d'un niveau de détail i et celle de son raffinement i+1 nécessite l'introduction d'éléments dits de continuité géométrique aux endroits où les objets sortent des faces détaillées.

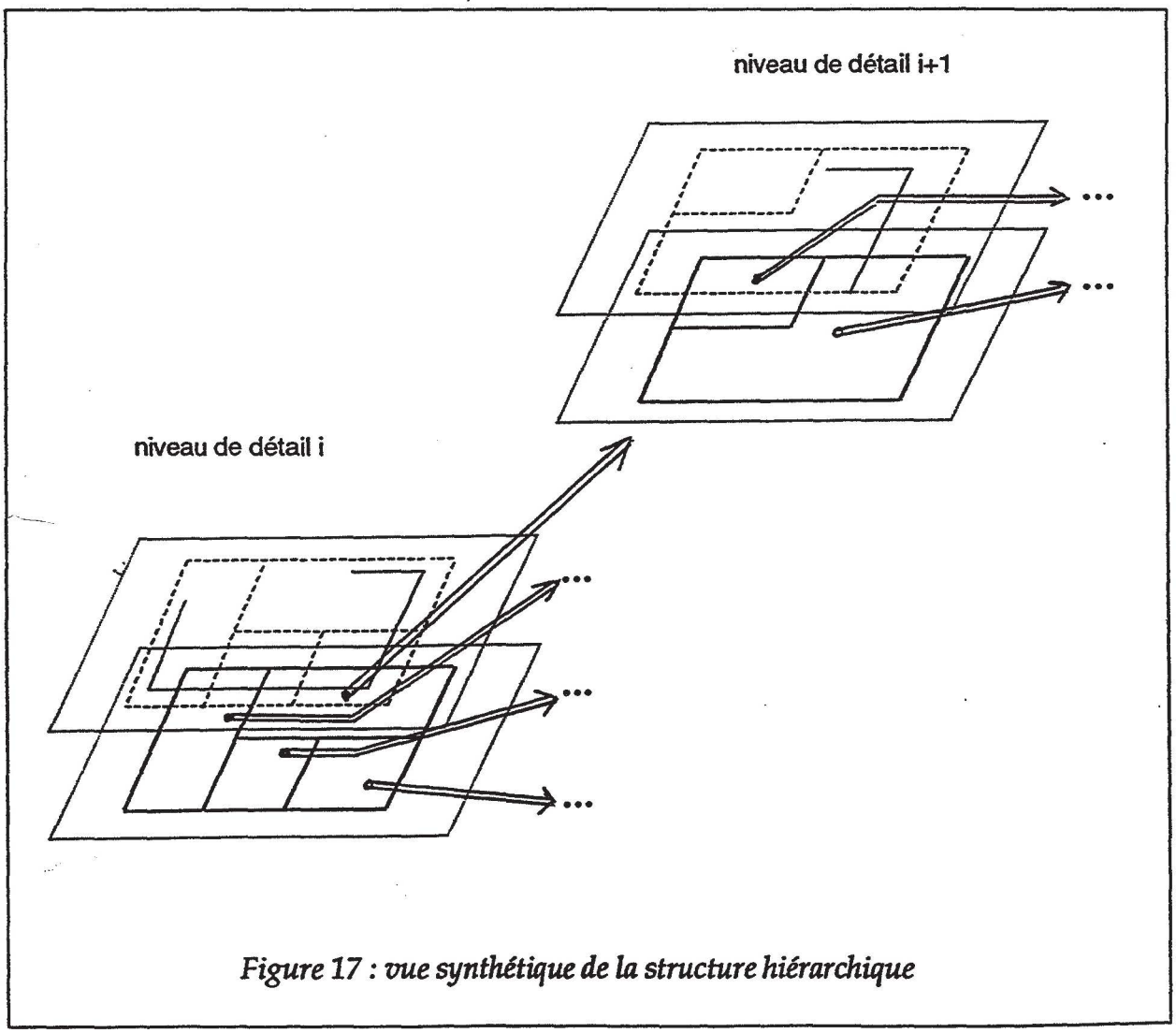


Figure 17 : vue synthétique de la structure hiérarchique

Dans ce qui suit, nous présenterons en section 2 une formalisation de la notion de hiérarchie et en section 3 ce que nous appelons les éléments de continuité géométrique. En section 4, nous présenterons la transformation dite "Zoom" permettant de spécifier des détails dans une description de scène donnée. Nous exposerons en section 5 les implications sur les liens typés 'détail' de l'ajout incrémental d'arêtes ou d'objets icôniques. En section 6, nous décrirons une opération dite d'exportation et ses implications sur les liens typés 'détail'. Enfin, nous présenterons en section 7 quelques opérations d'extension possibles, leurs implications sur les liens typés 'détail' et leur effet sur la stabilité du modèle.

2. FORMALISATION DE LA NOTION DE HIÉRARCHIE

Soit k>0 le nombre de sémantiques auxquelles appartiennent les objets composant une scène. Nous noterons **sb**, $0 \leq sb \leq k-1$, la sémantique privilégiée qui constitue également la sémantique de base pour la hiérarchie. Toute sémantique autre que **sb** sera appelée sémantique **auxiliaire**.

Un niveau plan au niveau de détail 0 sera représenté par **k** cartes planaires liées par des liens sémantiques : le tableau $\mathbf{tcarte[0 \ldots k-1]_0}$ avec $tcarte[i]_0$ de type ***struct cplanaire*** et 0 en indice rappelant le niveau de détail.

Ayant choisi une carte planaire comme base pour la hiérarchie, chaque face de cette carte planaire est propriétaire de tous les objets sémantiques qu'elle contient. Ces objets peuvent avoir n'importe quelle sémantique composant la scène : canalisations d'eau, circuits électriques, composants électroniques ... La description détaillée d'une face (ou plusieurs) d'une carte planaire représentant la sémantique de base **sb** à un niveau de détail i, $tcarte[sb]_i$, est assurée par **k** cartes planaires liées par des liens sémantiques : le tableau $\mathbf{tcarte[0 \ldots k-1]_{i+1}}$. Pour une sémantique x autre que sb, une description détaillée au niveau i+1, $tcarte[x]_{i+1}$, ne peut être définie que par rapport à une face ou plusieurs de la carte planaire de base parente de tous les niveaux i+1 : $tcarte[sb]_i$.

Comme le raffinement de la description d'une scène peut être itéré plusieurs fois, nous obtenons vite une structure hiérarchique assez complexe dans laquelle chaque noeud est composé de k cartes planaires superposées représentant différentes sémantiques. La figure 17 représente seulement une vue partielle d'une arborescence visualisant deux noeuds : une noeud parent et un de ses noeuds fils. Pour facilier la lisibilité de cette figure, nous n'y avons représenté que la sémantique de base et une seule sémantique auxiliaire. Le raffinement d'une partie du noeud parent est obtenu par extraction de tous les objets sémantiques contenus dans la face détaillée. La reproduction des objets extraits d'un noeud de détail i dans le raffinement i+1 pose le problème de la représentation multiple des objets. Il est donc nécessaire de maintenir liées les différentes représentations d'un même objet pour assurer une cohérence parfaite du modèle lors d'opérations élémentaires telles que l'ajout ou la suppression d'une arête à un certain niveau de détail.

2.1. Position du problème

Pour bâtir la hiérarchie nous avons donc besoin d'une opération d'initialisation qui réalise l'extraction de toutes les données sémantiques qui sont la 'propriété' d'une face détaillée. Cette opération doit reproduire les objets extraits de la face zoomée dans le niveau de raffinement supérieur et matérialiser le lien hiérarchique qui lie le niveau de détail parent et le niveau de détail fils. Cette initialisation duplique la représentation des objets physiques de la scène, elle doit donc compenser cela par l'établissement de liens typés 'détail' entre les différentes représentations d'un même objet physique.

L'aspect hiérarchique n'aurait pas d'intérêt s'il n'autorisait pas l'ajout d'arêtes au niveau de détail fils et donc de l'enrichir par rapport au niveau parent. L'opération d'ajout doit se faire tout en maintenant cohérent l'ensemble des représentations de la scène. Une arête d'une carte planaire pouvant appartenir à deux faces adjacentes, son éclatement peut induire des mises à jour dans les différents raffinements de ces faces adjacentes.

Enfin, le modèle doit permettre la remontée d'informations importantes d'un niveau de détail i vers son parent i-1. C'est ce que nous appelons l'opération d'exportation qui est indispensable. Elle permet par exemple de matérialiser au niveau parent le point où aboutit une arête d'un fils associé à une face F; cette matérialisation étant une aide nécessaire pour réaliser au niveau parent un dessin qui soit dans la continuité géométrique de celui du niveau fils (cf. figure 18).

Toutes ces opérations (initialisation / ajout / exportation) et leurs algorithmes doivent être maniées avec précaution dans le but de bâtir un modèle hiérarchique cohérent. Pour cela, nous nous sommes fixé deux principes que nous présentons dans les sections 2.3. et 2.4. L'initialisation doit respecter ces deux principes et les opérations d'ajout et d'exportation se doivent de les maintenir vérifiés pour tous les niveaux de raffinement existants de la scène modélisée.

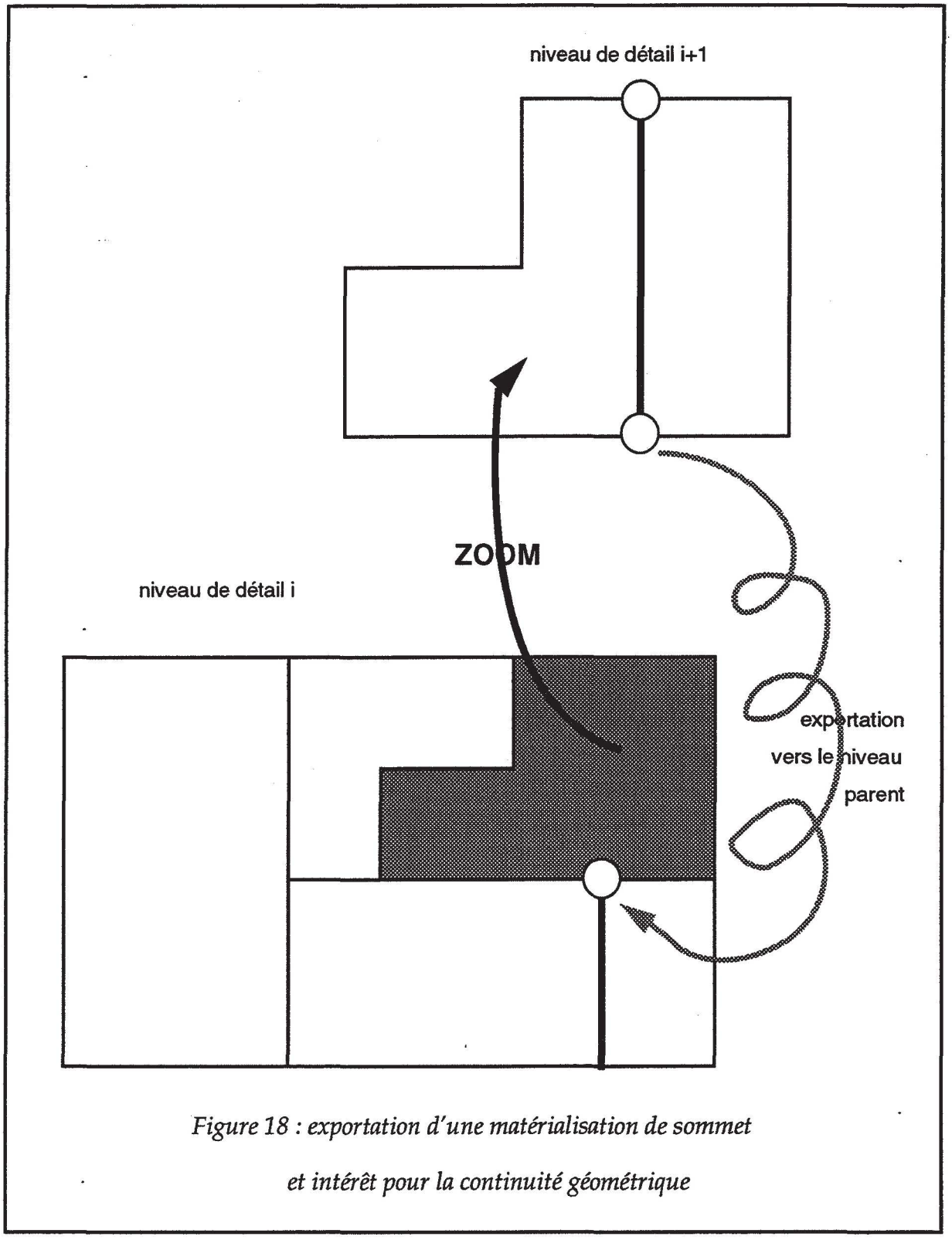


Figure 18 : exportation d'une matérialisation de sommet et intérêt pour la continuité géométrique

2.2. Notations

Cette convention de notation concerne les objets sémantiques composant la scène et figurant au niveau de raffinement i. Elle est destinée à clarifier le sens attribué aux indices dans les sections qui suivront :

- **obj_i** : le seul indice i indique le niveau de détail auquel appartient l'objet considéré et désigné par obj_i.

- **$obj_{h,i}$** : quand nous utilisons un double indiçage, le deuxième indice caractérise le niveau de détail auquel appartient l'objet considéré et désigné par $ob_{h,i}$; le premier indice étant utilisé pour distinguer l'objet considéré des autres objets figurant au même niveau de détail que lui.

2.3. Le premier principe de base pour le maintien de la cohérence

Pour maintenir la cohérence entre les différents niveaux de raffinement d'une scène décrite à l'aide de notre modèle hiérarchique, nous avons adopté le principe suivant :

soit obj un objet sémantique rectiligne, alors :

- le nombre d'arêtes le représentant dans les différents niveaux de détail doit évoluer dans le sens croissant en partant du niveau de détail 0 vers les niveaux supérieurs,

- une portion de l'objet obj, à un certain niveau de détail i, est délimitée soit par un objet icônique, soit par un autre objet du niveau i dont la représentation graphique n'est pas colinéraire à celle de obj,

- chaque portion de obj, à un certain niveau de détail i, est associée à un objet unique du niveau i qui est une représentaion détaillée exclusive d'un seul objet obj_{i-1} représentant obj dans le niveau i-1 (une portion de obj au niveau i ne doit pas contenir des parties de deux constituants ou plus de obj au niveau i-1),

- le modèle hiérarchique doit maintenir liés par des liens sémantiques portant le type 'détail' tous les objets sémantiques représentant le même objet physique dans différents niveaux de raffinement.

Figure 19 illustre une situation que n'autorise pas le principe ci-dessus. L'objet physique rectiligne qui y est considéré est vertical. Sa représentation graphique dans le niveau de détail i est assurée par les deux arêtes $ab_{1,i}$ et $ab_{2,i}$, et dans le niveau i+1 par $ab_{3,i+1}$ et $ab_{4,i+1}$. Cette situation n'est pas cohérente car $ab_{4,i+1}$ est un détail de $ab_{2,i}$ et d'une partie de $ab_{1,i}$.

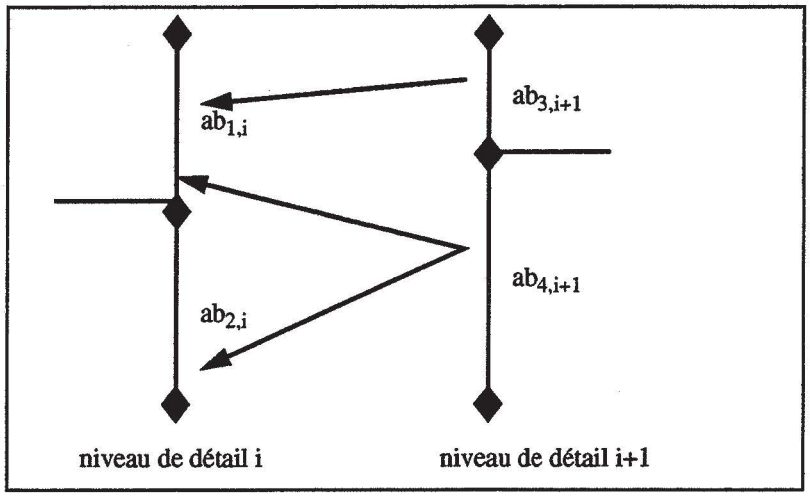


Figure 19 : cas non autorisé pour les liens sémantiques de détail

2.4. Le deuxième principe de base pour le maintien de la cohérence

Pour maintenir la cohérence entre les différents niveaux de raffinement d'une scène décrite à l'aide de notre modèle hiérarchique, nous avons adopté le second principe qui suit :

soit F une face d'une carte planaire de base de niveau de raffinement i à laquelle a été associé un niveau de détail i+1, alors :

- une transformation géométrique de type application affine doit permettre de transformer la face F en une face F' externe pour le niveau i+1, cette face F' étant la propriétaire de tous les objets sémantiques apparaissant au niveau de raffinement i+1,

- aucun ajout d'arête n'est autorisé à l'intérieur de la face F au niveau i; des compléments de description sont possibles au niveau i+h (h>1) : l'ajout d'arêtes n'est autorisé qu'en des feuilles de la hiérarchie,

- aucun ajout d'arête n'est autorisé à l'extérieur de la face F' au niveau i+1 (des compléments de description extérieurs à F' sont possibles au niveau i à l'intérieur de faces adjacentes à F, à moins que celles-ci ne soient elles mêmes zoomées. Dans ce cas, les ajouts sont à effectuer dans les descendants hiérarchiques de ces faces).

La figure 20 montre un exemple d'ajout d'arêtes incompatible avec le principe décrit ci-dessus.

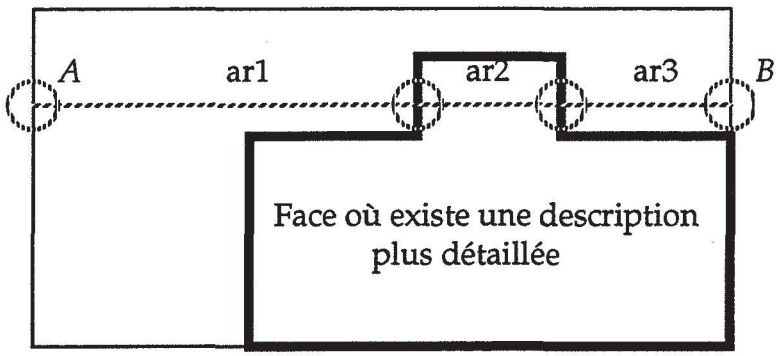


Figure 20 : refus de l'ajout du segment AB pour cause de non conformité avec le deuxième principe de la cohérence

2.5. LA STRUCTURE DE DONNÉES HIÉRARCHIQUE

Nous avons utilisé une structure de données hiérarchique qui s'inspire de l'arbre d'inclusion des faces. Si l'on considère un niveau plan donné, la structure de données qui le représente est la structure :

```
struct Liste_carte
    {
        struct cplanaire *tcarte[k];
        struct Liste_carte *plan_precedent;
        struct Liste_carte *plan_suivant;
        int numero_plan;
    }
```

Une personne modélisant une scène et souhaitant définir des niveaux de raffinement pour différentes parties du niveau plan considéré choisira des faces de la carte planaire de base pour obtenir une description faite de k cartes planaires associées à différentes sémantiques. Les différents détails associés à différentes parties d'un niveau plan seront accessibles par l'introduction dans la structure Liste_carte des champs *zoom_fils1* et *zoom_frere*. Le champ *zoom_pere* ajouté à cette structure permet d'accéder au niveau de détail parent. Les champs *zoom_pere, zoom_fils1* et *zoom_frere* sont de type **struct Liste_carte**. Un champ spécial de type **struct Idface_objets** permet la mémorisation des faces du niveau de raffinement parent ayant donné naissance au niveau de raffinement courant. Chaque structure Liste_carte est caractérisée par l'entier *niveau_zoom* qui indique le niveau de raffinement qu'elle représente. Enfin, le champ

Boîte_zoom permet d'établir le lien géométrique entre les objets appartenant à un niveau de détail i+1 et ceux qui les représentaient dans le niveau de détail parent i. *Boîte_zoom* est en fait la boîte rectangulaire englobant la face (ou les faces) détaillée au niveau i. La structure de données Liste_carte contenant la hiérarchie des niveaux de raffinement d'un niveau plan est la suivante :

```
struct Liste_carte
    {
        struct cplanaire *tcarte[k];
        struct Liste_carte *plan_precedent;
        struct Liste_carte *plan_suivant;
        struct Liste_carte *zoom_fils1;
        struct Liste_carte *zoom_frere;
        struct Liste_carte *zoom_pere;
        struct Idface_objects *faces_pere;
        int numero_plan;
        int niveau_zoom;
        struct rec_englob boîte_zoom;
    }
```

3. LES ÉLÉMENTS DE CONTINUITÉ GÉOMÉTRIQUE

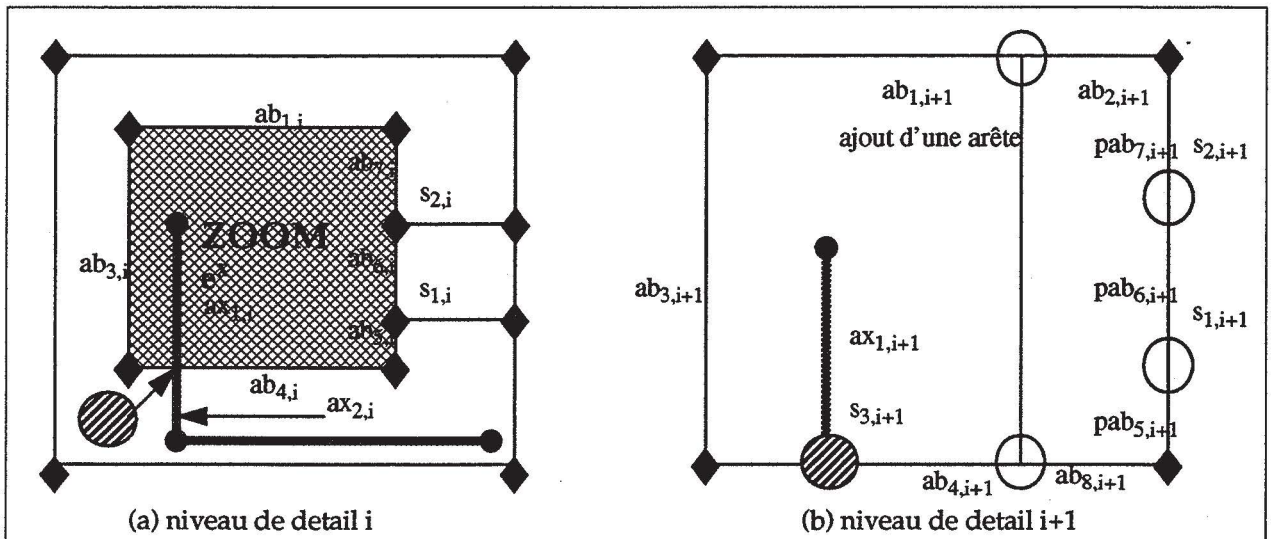


Figure 21 : transmission de la propriété du père vers le fils

La figure 21(a) est la description d'une scène à un certain niveau de détail i. La carte planaire représentant la sémantique de base est visualisée par des segments reliant des sommets (losanges). Parmi les objets sémantiques que cette carte planaire modélise figurent $ab_{1,i}$, $ab_{4,i}$ et

$ab_{6,i}$, ab rappelant que l'arête appartient à la carte planaire de base. Une sémantique autre que celle de base (sémantique auxiliaire) est représentée par une carte planaire tcarte[x] visualisée par des segments épais texturés reliant des sommets (petits disques). Parmi les objets modélisés par cette carte planaire figure $ax_{1,i}$ (arête e^x verticale), ax rappelant que l'arête appartient à une carte planaire de **auxiliaire**.

La notion de sémantique de base pour la hiérarchie est synonyme de celle de propriété. Or, pour pouvoir établir clairement la liste des objets sémantiques qui sont la propriété d'une face "Zoomée" (la face texturée de la figure 21(a) par exemple), les objets ayant une sémantique de auxiliaire doivent être coupés symboliquement à la traversée de la face zoomée. Pour pouvoir effectuer une coupe symbolique de ces objets ($ax_{1,i}$ en $ax_{1,i}$ et $ax_{2,i}$), et pour pouvoir les reconstituer, nous avons mis en place des éléments de continuité à l'endroit de la coupe (disque texturé de la figure 21(a)). Un élément de continuité géométrique, noté ecg par la suite, est en fait un objet icônique qui, par son insertion au niveau de détail i, provoque la coupure du segment ax_i sur lequel il est installé. Dans le cas de la figure 21(a), l'insertion d'un ecg entraine la création d'un nouveau sommet dans la carte planaire représentant la sémantique de $ax_{1,i}$, objet sémantique traversant la face "Zoomée". La reconstitution des objets comme $ax_{1,i}$ coupés symboliquement sera possible si les deux objets $ax_{1,i}$ et $ax_{2,i}$ sont liés sémantiquement à l'objet ecg par des liens typés (cf. section 2.1. de la partie B de ce chapitre).

Nous avons voulu également intégrer dans notre modèle des moyens pour assurer la continuité géométrique entre les dessins de scène réalisés à différents niveaux. Avant de rajouter un quelconque détail, la face texturée de la figure 21(a) possèdera une description détaillée de sa sémantique de base représentée par une carte planaire où ne figurent que quatre arêtes. En effet, les algorithmes de construction des cartes planaires fusionnent les arêtes opposées. $s_{1,i+1}$ et $s_{2,i+1}$ ne sont donc pas de vrais sommets au sens des cartes planaires : ils n'ont aucune existence géométrique. Pour permettre de continuer, au niveau de détail i+1, les arêtes horizontales aboutissant au niveau de détail i aux sommets $s_{1,i}$ et $s_{2,i}$, nous avons mis en place d'autres éléments de continuité géométrique. Ces ecg sont représentés figure 21(b) par des disques non texturés en $s_{1,i+1}$ et $s_{2,i+1}$. Leur création se fera donc en des sommets marqués empêchant ainsi toute fusion de segments colinéaires y aboutissant. Un utilisateur du modèle pourra donc facilement utiliser ces sommets, marqués par des icônes, pour réaliser au niveau de détail i+1 des dessins qui assurent la continuité géométrique avec ceux du niveau de détail i.

L'ajout de ces éléments de continuité géométrique permet en plus de satisfaire le premier principe de cohérence de la hiérarchie (cf. section 2.3. de cette partie). Inversement, toute arête ajoutée au niveau de détail i+1 entrainera, en chacune de ses intersections avec les arêtes délimitant les objets sémantiques du niveau de détail i+1, la création d'un élément de continuité géométrique (cf. figure 21(b)).

Dans la gestion de la hiérarchie de notre modèle, nous exportons automatiquement les éléments de continuité créés dans une description donnée vers toutes les descriptions détaillées filles. L'export de ces éléments dans le sens inverse est commandé par l'utilisateur.

3.1. La sémantique induite par les éléments de continuité géométrique ayant la sémantique de base : ecgsb

Dans les cartes planaires ayant la sémantique de base sb, nous distinguons les sommets ayant un sens physique (ou géométrique) des autres sommets. Nous dirons qu'un sommet a un sens physique si deux arêtes non colinéaires au moins aboutissent en ce sommet. Les algorithmes de construction de carte planaire fusionnent les arêtes opposées; ils ne produisent en principe que des sommets ayant un sens physique. Or, si nous voulons, comm sur la figure 21(b), avoir des cartes planaires où figurent des sommets semblables à $s_{1,i+1}$ et $s_{2,i+1}$ qui n'ont pas de sens physique, nous devons donner à ces sommets un certain sens en y installant des icônes d'ecg par exemple.

Une arête de la carte planaire de base peut donc, au regard de ses deux sommets, être associée soit à un objet noté ab pour rappeler qu'il est associé à une arête physique de la sémantique de base, soit à un objet noté pab (partie d'un objet ab). D'un point de vue sémantique, nous ne gardons l'association arête-ab que pour des arêtes dont les deux sommets ont un sens physique dans la carte planaire de base courante. Dans le cas où un des deux sommets de l'arête n'a pas de sens physique dans la carte planaire courante, le sommet en question est éliminé par fusion des arêtes colinéaires incidentes à moins qu'un élément **ecgsb** ne soit placé en ce sommet (cf. figure 22). Chaque arête de la carte planaire de base ne pouvant être associée à un objet de type sémantique ab est alors associée à un objet pab lié par un lien typé 'partie_de' à un autre objet de type sémantique ab. Dans ce cas, la représentation graphique de ab est la succession des arêtes colinéaires comprenant l'arête-pab en question et s'arrêtant en deux sommets ayant un sens physique pour la carte planaire courante.

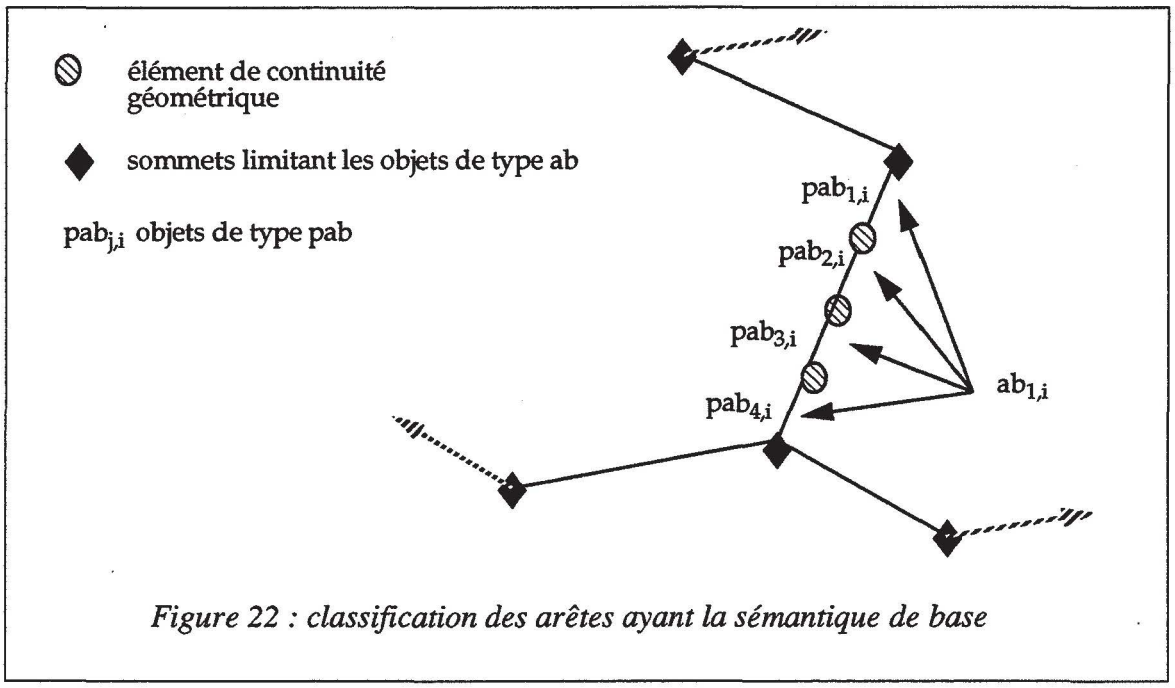


Figure 22 : classification des arêtes ayant la sémantique de base

3.2. La sémantique induite par les éléments de continuité géométrique auxiliaires

Les objets appartenant à chaque sémantique x, différente de celle de base, constituent des réseaux (de distribution d'électricité, de canalisations d'eau, de circuits de chauffage ...). Dans notre modélisation, nous ne prenons pas en compte le cas où les sémantiques auxiliaires sont représentées par des cartes planaires ayant des faces internes. Ces réseaux (cartes planaires auxiliaires) sont donc constitués principalement de lignes séparées par des objets icôniques de connexion XCON. Chaque objet-ligne, noté désormais **l_ax**, est décomposé en un (ou plusieurs) objet 'partie_de_ligne' noté **ax** associé à une arête auxiliaire. Chacun des objets ax est lié à l'objet l_ax dont il constitue une partie par un lien sémantique typé 'partie_de' (cf. section 2.1. de la partie B). Les objets icôniques de connexion XCON ont la spécificité de couper l'objet ax auxiliaire sur lequel ils sont installés, et d'être connectés sémantiquement aux deux partie_de_ligne ax qui lui sont incidentes. Un exemple d'objet icônique XCON est l'objet matérialisant un élément de continuité géométrique auxiliaire **ecgsx**.

Les objets auxiliaires dont la représentation graphique est faite d'arêtes (cf. figure 27) sont donc de deux types : les lignes et les partie_de_ligne. Leur création obéit aux règles suivantes :

- la création d'un objet de type ligne (l_ax) se fait dès la détection d'un chemin (succession d'arêtes) identifié par une liste de sommets dont seuls le premier et le dernier sont dun des types suivants :

  1- **SR1**, sommet auxiliaire duquel ne part qu'une seule arête,

  2- **SR2**, sommet auxiliaire en lequel est placée une icône XCON.

- à chaque arête est associé un objet de type partie_de_ligne (ax), qui est lié à l'objet de type ligne (l_ax) qui l'emprunte par un lien typé 'partie_de'.

4. LA TRANSFORMATION "Zoom"

4.1. Initialisation de l'opération "Zoom"

Au lancement de l'opération "Zoom", nous avons besoin de calculer les paramètres qui nous permettront d'établir un lien géométrique entre les données de la partie de la carte planaire de base à détailler et celles des cartes planaires qui les détaillent. Ce lien géométrique est une application affine qui permet de calquer, après transformation, la boîte rectangulaire englobant l'ensemble des faces à détailler sur une zone rectangulaire de l'écran d'un ordinateur où seront visualisées toutes les cartes planaires de détail. L'initialisation de l'opération "Zoom" s'effectue en trois phases.

En phase 1, nous calculons la boîte ENGLOB englobant les boîtes englobantes des faces sélectionnées en vue de l'opération "Zoom".

En phase 2, nous séparons les données qui seront détaillées dans la carte planaire de détail, de celles qui ne le seront pas. En ce qui concerne les données de la sémantique de base représentées par des arêtes, cela est simple puisque seules les arêtes participant à la représentation graphique des faces à détailler feront partie des données à détailler. Par contre, le problème est moins simple pour les données auxiliaires représentées par des arêtes. De ce fait, nous scrutons tous les objets auxiliaires représentés graphiquement par des arêtes et nous testons si ils traversent ou non des faces à détailler. Pour effectuer la séparation, nous introduisons les éléments de continuité géométrique auxiliaires. Ce sont des objets icôniques qui sont en fait une extension des objets icôniques de connexion auxiliaires XCON. La deuxième phase de l'initialisation de l'opération "Zoom" consiste donc en l'installation de ces objets en des endroits où les lignes auxiliaires traversent les faces sélectionnées en vue de l'opération "Zoom".

La phase 3 de l'initialisation de l'opération "Zoom" consiste simplement à créer la structure Liste_carte qui va contenir la description détaillée des faces sélectionnées, à affecter les pointeurs la situant dans la hiérarchie par rapport à la Liste_carte parente et ses filles antérieurement spécifiées et à affecter le champ boîte_zoom de la nouvelle Liste_carte à partir de ENGLOB.

4.2. La transformation géométrique "Zoom"

La transformation géométrique de l'opération "Zoom" permet d'associer, de façon bijective, à chaque point de la carte planaire parente, un point de la carte planaire fille. L'ensemble des faces sélectionnées en vue de l'opération "Zoom" est englobé par la structure *boîte_zoom* (cf. section 2.5. de cette partie). *boîte_zoom* est définie par son sommet haut gauche (xb,yb) et ses longueur et largeur wb et hb.

Soit (XZOOM, YZOOM) le point définissant le sommet haut gauche d'un rectangle RECT occupé entièrement par l'ensemble des faces internes "Zoomées" dans la carte planaire fille. Soient WZOOM et HZOOM respectivement les longueur et largeur de RECT. XZOOM, YZOOM, WZOOM ET HZOOM sont des constantes.

Soient M(x,y) un point de la carte planaire parente et N(x1,y1) son transformé dans la carte planaire fille. x1 et y1 sont fournis par les expressions suivantes:

$$x1 = (WZOOM * (x - xb)/wb) + XZOOM$$

$$y1 = (HZOOM * (y - yb)/hb) + YZOOM$$

La transformation géométrique inverse existe, en raison de la non nullité de WZOOM et HZOOM.

4.3. La sémantique de la hiérarchie mise en place dès le lancement de l'opération "Zoom" - algorithmes

Comme nous l'avons dit dans la section 2. de cette partie, pour une sémantique donnée, la description détaillée d'une partie d'une carte planaire de niveau de détail i est représentée par une autre carte planaire de niveau de détail i+1. Or, dans la structure de données carte planaire, nous associons à chaque arête un objet sémantique. Il en découle qu'un objet physique appartenant à un niveau plan de la scène à modéliser peut avoir autant d'objets sémantiques qui le représentent que de niveaux de détail spécifiés dans ce plan. Le problème de la représentation multiple d'un même objet se pose donc. Pour le résoudre, nous proposons de lier, par un lien typé 'détail' (cf. section 2.1. de la partie B), les représentations du même objet dans une carte planaire et dans ses cartes planaires filles.

Dans la suite de cette section, nous allons décrire comment nous mettons en place les liens 'détail' dès le lancement de l'opération "Zoom". L'initialisation traitée dans la section 4.1. crée l'enregistrement Liste_carte qui contiendra les cartes planaires associées à toutes les sémantiques pouvant constituer une description détaillée. Maintenant, nous nous occupons de reproduire dans Liste_carte les objets sémantiques qui sont la propriété des faces détaillées. Notre idée directrice, pour la mise en place des liens 'détail', consiste à profiter de la structure des données dans le niveau parent pour lier chaque objet créé dans le niveau de détail fils à son homologue dans le niveau parent. Nous allons présenter cela, cas par cas, selon la nature sémantique des objets à reproduire.

4.3.1. Les objets associés à la sémantique de base

Les arêtes qui constituent les faces de la carte planaire de base intervenant lors du lancement de l'opération "Zoom" sont bien structurées en termes de carte planaire au niveau de détail parent. Soient fz la transformation géométrique définissant l'opération "Zoom" à effectuer et LPT la liste des points images des extrémités des arêtes permettant la représentation graphique des faces "zoomées". La construction de la carte planaire à partir de la liste de points LPT ne fait appel à la procédure de fusion d'arêtes qu'en des points précis : les points d'où partent, au niveau parent, des arêtes extérieures aux faces zoomées (les sommets $s_{1,i}$ et $s_{2,i}$ de la figure 21). En empêchant la fusion d'arêtes par un indicateur, positionné dès le lancement de l'opération "Zoom", nous construisons une carte planaire fille, ayant la sémantique de base, dont les faces internes (faces), hormis la racine de l'arborescence des faces, peuvent être identifiées, arête par arête, à celles spécifiées lors de l'opération "Zoom".

Il est important de signaler qu'en vue de la représentation graphique d'une face F, nous n'utilisons pas seulement les arêtes qui permettent son parcours, mais aussi toutes celles des faces externes Fe_m filles de F dans l'arborescence des faces (cf. figure 23). Les arêtes des faces externes Fe_m sont donc prises en compte lors du calcul des points constituant LPT.

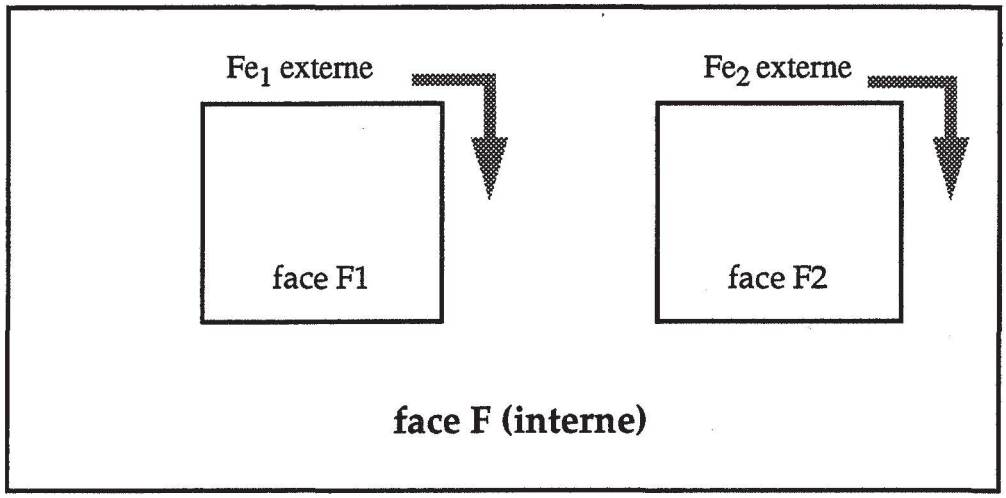


Figure 23 : représentation graphique de la face F par ses arêtes en tant que face interne et par les arêtes de ses filles externes Fe_1 et Fe_2

L'algorithme de mise en place des liens 'détail' relatifs aux objets créés dans la nouvelle description détaillée et associés aux arêtes et aux éléments de continuité ayant la sémantique de base peut se résumer ainsi. Pendant la construction de la carte planaire de base fille à partir de la liste de points LPT, nous créons un élément de continuité géométrique ecgsb en chaque sommet où la procédure de fusion d'arêtes doit être déclenchée. Lors de la création d'un ecgsb dans la carte planaire fille, un ecgsb est créé dans le niveau parent à l'endroit du transformé "Zoom" inverse de l'ecgsb (sauf s'il en existait déjà un). L'ecgsb créé dans le niveau parent est lié par un lien typé 'détail' à son homologue dans le niveau de détail courant. Quand la création de la carte planaire fille est terminée, nous mettons en place les liens détail relatifs aux objets sémantiques représentés par des arêtes. Ceci est effectué en cherchant à identifier les objets-arêtes participant à la représentation graphique des faces zoomées. Or, nous avons vu dans le premier chapitre que chaque face est caractérisée par son brin de naissance. Il nous suffit donc d'identifier, comme dans l'algorithme de construction incrémentale de la carte planaire globale (cf. chapitre I), chaque face interne zoomée et chacune des filles externes des faces zoomées avec une face de même sens dans l'arborescence des faces de la carte planaire fille. Grâce au procédé d'antifusion, des liens typés détail peuvent être mis en place en parcourant, de façon coordonnée, les arêtes-objets de deux faces identifiées (parente et fille) à partir de leurs brins de départ. Un dernier élément doit cependant être signalé : la construction d'une carte planaire associe, dans un premier temps, un objet de type ab à chaque arête possédant la sémantique de base. Un parcours des sommets de la carte planaire fille, selon l'ordre lexicographique, permet de rectifier la sémantique. Les sommets où cette rectification doit intervenir sont marqués par des icônes ecgsb. Soient AR1 et AR2 les deux arêtes de part et d'autre du sommet Som et OBJ1 et OBJ2 les objets sémantiques qui leurs sont associés. La rectification de la sémantique est effectuée par l'algorithme qui suit :

début

- cas (OBJ ET OBJ1 sont de type ab)

  changer leur type en pab
  créer OBJ2, nouveau objet ab sans représentation graphique
  lier OBJ à OBJ2 par un lien typé 'partie_de'
  lier OBJ1 à OBJ2 par un lien typé 'partie_de'

- cas (OBJ est de type ab ET OBJ1 est de type pab)

  changer le type de OBJ en pab
  chercher OBJ2 lié à OBJ1 par un lien typé 'partie_de'
  lier OBJ à OBJ2 par un lien typé 'partie_de'

- cas (OBJ1 est de type ab ET OBJ est de type pab)

  effectuer le même traitement que dans le cas précédent en permutant les rôles de OBJ et OBJ1.

- cas (OBJ ET OBJ1 sont de type pab) il n'y a pas de rectification à faire.

fin

4.3.2. Les objets associés à une sémantique auxiliaire

Dans la section 4.1., nous avons montré comment nous plaçons les objets icôniques symbolisant les éléments de continuité auxiliaires aux traversées des frontières des faces sélectionnées en vue de l'opération "Zoom". Leur installation est effectuée pendant la phase d'initialisation de l'opération "Zoom". La mise en place des liens typés 'détail' relatifs aux objets auxiliaires passe par trois étapes :

1- parcourir les objets auxiliaires de la carte planaire parente et considérer uniquement ceux dont le type est ax selon le sens de la section 3.2. de cette partie. Chacun des objets considérés est associé à une seule arête auxiliaire. Dès que l'arête associée à un objet auxiliaire est entièrement à l'intérieur de l'une des faces zoomées, rajouter l'image de cette arête par la transformation "Zoom" dans la structure fille : carte planaire auxiliaire de même sémantique que l'objet courant,

2- parcourir les objets auxiliaires de la carte planaire parente, qui sont associés à une icône. Dès qu'une icône est à l'intérieur ou sur la frontière de l'une des faces zoomées faire : rajouter l'image de cette icône par la transformation "Zoom" dans la liste des objets auxiliaires de la carte planaire fille si elle n'y existait pas, en établissant un lien typé 'détail' entre l'icône et son image dans la carte planaire fille.

3- pour chaque objet OBJ de type ligne (l_ax) de la carte planaire parente, chercher son homologue OBJZ de type ligne dans la carte planaire fille et le lier à lui par un lien typé 'détail'. L'identification d'un homologue OBJZ d'un objet OBJ de type ligne s'effectue par un parcours des parties POBJ de OBJ. Pendant ce parcours, chaque partie POBJ doit pouvoir être identifiée à une partie POBJZ de OBJZ. Si tel est le cas, POBJ est lié par un lien typé 'détail' à POBJZ.

5. MAINTIEN DES LIENS TYPÉS 'DÉTAIL' EN CAS D'AJOUT DE SEGMENTS OU D'ICÔNES À UN NIVEAU DE DÉTAIL DONNÉ

5.1. Test et traitement préliminaire des segments ou icônes ajoutés

Nous nous plaçons dans la description d'un niveau plan de la scène à modéliser où les objets associés aux arêtes ont des liens typés 'détail' avec ceux d'une carte planaire parente ou d'une carte planaire fille. Etant donnés les traitements précédemment décrits, les cartes planaires représentant la description courante sont toutes dans un état de construction incrémentale. Un segment à ajouter au dessin courant de la scène est caractérisé par ses deux points extrémités. De même, un objet icônique est caractérisé par le point représentant l'endroit où nous désirons placer son centre (centre de sa boîte englobante).

Pour maintenir la cohérence entre les diverses données dans les différents niveaux de détail, nous devons tout d'abord être en accord avec le second principe de la cohérence (cf. section 2.4. de cette partie). Lorsqu'il s'agit de l'ajout d'une icône, nous arrivons sans difficulté à savoir si son emplacement se situe à l'intérieur d'une zone d'incohérence. Par contre, une connaissance de ce type est moins simple à acquérir quand il s'agit de l'ajout d'un segment. Considérons le cas de la figure 20 où l'on ajoute un segment *AB* (représenté en pointillé) à une carte planaire de niveau de détail i dont une face (représentée en gras) possède une description de niveau de détail i+1. Nous découpons les segments ajoutés aux frontières des faces délimitant le niveau de détail courant par rapport à la carte planaire parente, et aux frontières des faces délimitant les niveaux de détail fils par rapport à la carte planaire courante. Ceci permet de décider si l'ensemble des points définissant les segments ajoutés induit des conflits dans les descriptions des niveaux de détail. Dans le cas de la figure 20, le traitement du segment *AB* passe par celui des arêtes ar1, ar2 et ar3 : l'arête ar2 est rejetée pour cause de conflit avec le niveau de détail existant i+1, et cela provoque le rejet de l'ensemble des segments ajoutés.

Les traitements décrits dans les paragraphes qui suivent sont faits pendant la construction des structures cartes planaires locales (cf. chapitre I) en chacun des nouveaux sommets à insérer (points saisis ou points d'intersection entre les nouveaux segments saisis et les arêtes pré-existantes de la carte planaire). Les cartes planaires locales des sommets étant mises à jour en respectant l'ordre lexicographique, nous ne nous occuperons, pour chaque arête ajoutée, que d'un sommet à la fois dans les algorithmes qui suivent.

5.2. Ajout d'arêtes à une carte planaire ayant la sémantique de base

Les arêtes ajoutées qui pourraient être concernées par une mise à jour des liens 'détail' sont celles qui interviennent, d'une manière ou d'une autre, sur des arêtes ayant déjà des liens 'détail'. Donc, l'ajout d'une arête isolée du reste du dessin d'un niveau de détail, si cet ajout satisfait les règles de cohérence, ne met pas en cause les liens typés 'détail'. De ce fait, parmi tous les ajouts d'arête acceptables deux cas seulement nous intéressent :

- un des sommets de l'arête ajoutée est celui d'une arête pré-existante ayant des liens typés 'détail'. Dans ce cas, l'arête ajoutée ne provoque pas d'éclatement d'arête,
- un des sommets de l'arête est strictement à l'intérieur d'une arête pré-existante ayant des liens typés 'détail'. Dans ce cas, l'arête ajoutée provoque un éclatement d'arête.

Nous allons donc traiter successivement ces deux cas.

5.2.1. Ajout d'une arête ne provoquant pas d'éclatement d'arête

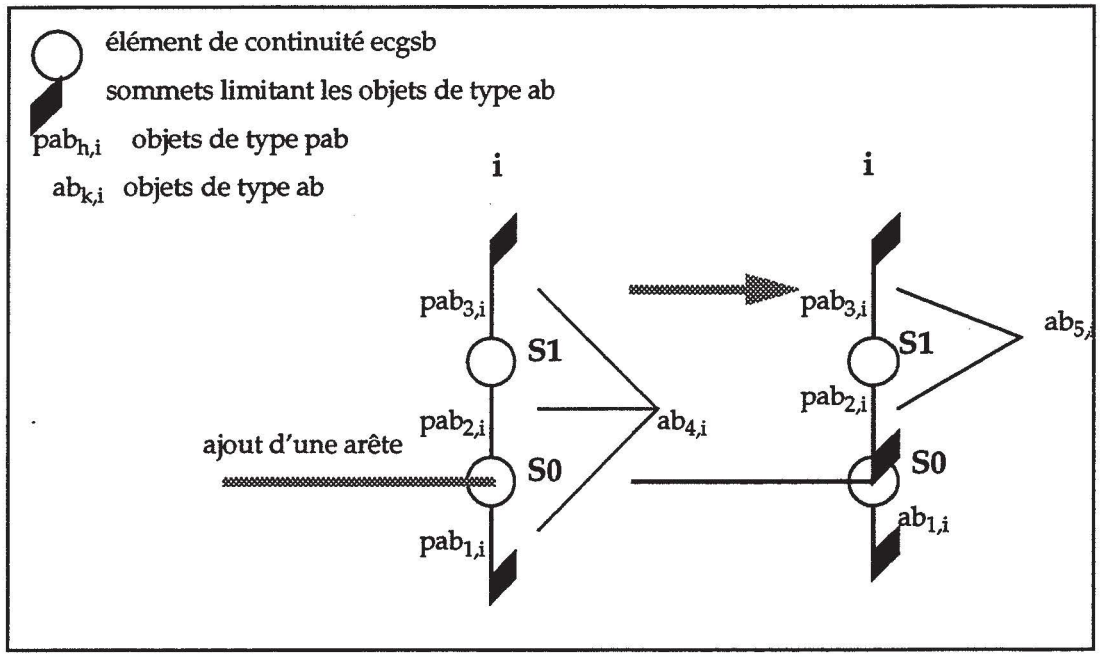


Figure 24 : ajout d'une arête dont un des sommets est confondu avec un sommet existant

Soit AR une arête ajoutée à une carte planaire CPB possédant la sémantique de base, de **niveau de détail i**, et telle que un de ses sommets, **S0**, soit celui d'arêtes pré-existantes ayant déjà des liens typés 'détail'. Si **S0** avait déjà un sens physique dans CPB, alors les objets sémantiques pré-existants ne changent pas et leurs liens non plus. Sinon, l'ajout de AR dans la structure de carte planaire locale à **S0** peut attribuer à ce sommet le sens physique qu'il ne possédait pas. Il en découlerait un changement d'objets de type pab en objets de type ab (cf. section 3.1).

Considérons le cas de la figure 24. Au point où nous intervenons dans la construction de la carte planaire locale, nous savons que le nombre des arêtes incidentes en **S0** a changé. L'entrée de notre algorithme se fait en **S0**. Nous considérons les arêtes qui sont incidentes en ce sommet. Pour modifier les objets existants au voisinage de **S0**, nous nous servons de *ar* arête incidente en **S0** qui nait en **S0**. Ceci tient compte du fait qu'il suffit de connaître une partie $pab_{j,i}$ d'un objet ab pour en déduire tous les objets $pab_{k,i}$ qui le constituent. *ar* est associée à un objet de type $pab_{2,i}$ constituant une partie d'un objet $ab_{4,i}$. Soient **S1** l'autre sommet de *ar*, **ar1** l'arête incidente à **S1** et **ar2** l'arête qui suit ar1 dans le σ-ordre autour de **S1.**

L'algorithme de mise à jour des liens typés 'détail' lors de l'ajout de l'arête AR repose sur les remarques suivantes (cf. figure 24) :

- un objet de type ab composé de plusieurs pab ne possède des liens 'détail' que par l'intermédiaire de ses parties pab,

- nous ne devons garder d'objet de type ab composé de parties pab que si le nombre de ses parties est supérieur à 1,

- si l'on considère les arêtes dont les deux sommets sont inférieurs ou égaux, en ordre lexicographique à **S0**, il en existe au moins une associée à un objet de type pab partie de ab. Dans le cas de la figure 24 où ce pab est unique, il suffit de supprimer le lien 'partie_de' qui le liait avec l'objet ab et de changer son type sémantique de pab à ab. Ses liens 'détail' restent alors valides.

- si l'on considère les arêtes dont les deux sommets sont supérieurs ou égaux, en ordre lexicographique à **S0**, nous pouvons n'avoir au départ qu'une seule arête associée à un objet pab partie de ab (cas où **S1** possède un sens physique). Dans ce cas, nous changeons bien sûr le type sémantique de ce pab en ab. Ses liens 'détail' restent alors valides.

5.2.2. Ajout d'une arête provoquant l'éclatement d'une arête existante

Supposons que l'**ajout de** l'arête **ar1** se fasse **au niveau de détail i** et provoque l'éclatement d'une arête *ar* ayant des liens 'détail'.

Etant donné le deuxième principe de la gestion de la cohérence, nous déduisons deux choses de l'acceptation de l'ajout de ar1 : premièrement, ar1 étant acceptée en étant située d'un

côté par rapport à *ar*, un niveau de détail i+1 bordé par *ar* est forcément situé de l'autre côté; deuxièmement, il ne peut exister qu'une seule carte planaire de niveau de détail i+1 liée à *ar*.

Soient (x,y) le point de *ar* en lequel intervient l'éclatement et obj_i l'objet sémantique associé à *ar*. Soient **S0** et **S1** les deux sommets de *ar* rangés selon l'ordre lexicographique.

L'initialisation de l'opération "Zoom" associe à chaque arête du niveau de détail i-1 exactement une arête dans le niveau de détail i (premier principe de la cohérence). Si nous concevons l'ensemble des algorithmes de mise à jour des liens typés 'détail' en respectant le premier principe de la cohérence, alors, nous pouvons déduire une donnée importante de notre raisonnement :

> il ne peut y avoir qu'un seul objet du niveau de détail i-1 lié par un lien typé 'détail' à l'objet obj_i avant l'ajout de ar1.

On a donc le cas représenté sur la figure 25, avec les liens 'détail' antérieurs à l'ajout d'une arête, et l'ensemble des liens à mettre à jour suite à cet ajout.

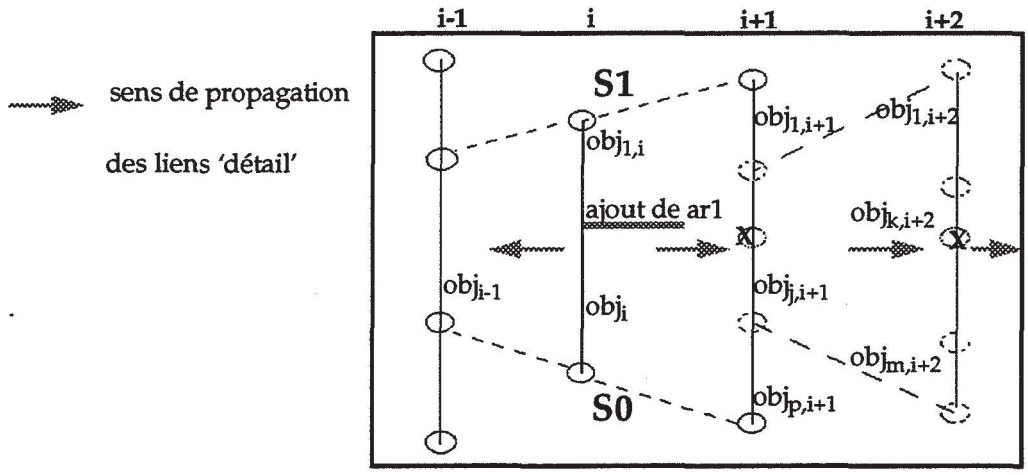


Figure 25 : éclatement de l'arête associée à obj_i et chaîne des mises à jour de ses liens typés 'détail'

L'algorithme de propagation des mises à jour des liens 'détail' vers les descendants du niveau i et vers la carte planaire parente est basé d'abord sur la procédure récursive recursif_lien_descendant. Cette procédure porte le point d'impact (x,y) de l'arête ajoutée aux emplacements géométriques qui lui correspondent dans tous les niveaux de détail plus élevés. Il en résulte des éclatements d'arêtes dans tous les niveaux de détail i+k qui existent. Cet éclatement d'arêtes dans les descendants a pour but de respecter le premier principe de la cohérence en associant une arête du niveau de détail i+k+1 à une seule arête du niveau de détail parent i+k. L'intérêt du premier principe de cohérence que nous nous sommes fixé est d'offrir la possibilité de

maintenir cohérente notre hiérarchie avec des algorithmes bien paramétrés grâce à cette unicité. On peut facilement observer que sans cet éclatement, $obj_{j,i+1}$ de la figure 25 devrait être associé à deux arêtes représentant respectivement les objets obj_j et $obj_{1,i}$

La récursivité de recursif lien descendant fait qu'elle commence d'abord par éclater l'arête correspondant à celle éclatée dans le niveau i, et associée à $obj_{k1,f}$ qui se situe dans une feuille de l'arborescence. Un parcours récursif nous fait arriver à $obj_{k1,f}$ via un paramètre de la procédure récursive qui est par exemple $obj_{k2,f-1}$ anciennement lié par un lien typé 'détail' à $obj_{k1,f}$. Avant de remonter dans la recursion au niveau de détail f-1, nous marquons haut toutes les arêtes du niveau f qui étaient liées par des liens 'détail' à $obj_{k2,f-1}$ et qui sont au dessus du point d'impact (x_f,y_f) de l'arête ajoutée, au niveau de détail f. Nous profitons de notre passage par la procédure de construction de carte planaire locale au nouveau sommet (x_f,y_f) pour mettre à jour le type sémantique de $obj_{k1,f}$ qui passerait éventuellement du type ab au type pab et sera éclaté en $obj_{k1,f}$ et $obj_{k3,f}$, et également pour créer en (x_f,y_f) un élément de continuité géométrique ecgsb s'il n'en existait pas en ce point. Une fois remonté dans la hiérarchie au niveau f-1, nous pouvons éclater $obj_{k2,f-1}$ en deux objets $obj_{k2,f-1}$ et $obj_{k4,f-1}$. Soit $\mathbf{obj_{h,f-1}}$ celui des deux dont l'arête associée a un deuxième sommet supérieur lexicographiquement au point d'impact (x_{f-1},y_{f-1}). De même, nous assurons l'adéquation des types sémantiques de $obj_{k2,f-1}$ $obj_{k4,f-1}$ en installant éventuellement une nouvelle icône ecgsb en (x_{f-1},y_{f-1}) que nous lierons à celle remontée du niveau f installée en (x_f,y_f). Contrairement au niveau de la feuille f, nous effectuons au niveau f-1 et tous les niveaux f-h qui suivront dans la récursion le traitement qui se schématise dans le passage de f vers f-1 par ce qui suit :

(I) lier $\mathbf{obj_{h,f-1}}$ par des liens typés 'détail' à ceux du niveau f, qui étaient liés par un lien 'détail' à $obj_{k2,f-1}$ et qui ont été marqués *haut* par recursif lien descendant; annuler leur lien avec $obj_{k2,f-1}$. L'annulation des marquages de tous les objets $obj_{k,f}$, liés à $obj_{k2,f-1}$, est faite pendant le parcours des liens 'détail' de $obj_{k2,f-1}$; les objets anciennement liés à $obj_{k2,f-1}$et marqués *bas* restent liés à $obj_{k2,f-1}$,

Figure 26 représente le traitement (I) entre les niveaux de détail j et j-1, i<j<f+1, avec i niveau de détail où interviennent l'ajout d'une arête et l'éclatement de *ar* et f le niveau le plus imbriqué des représentations de *ar*.

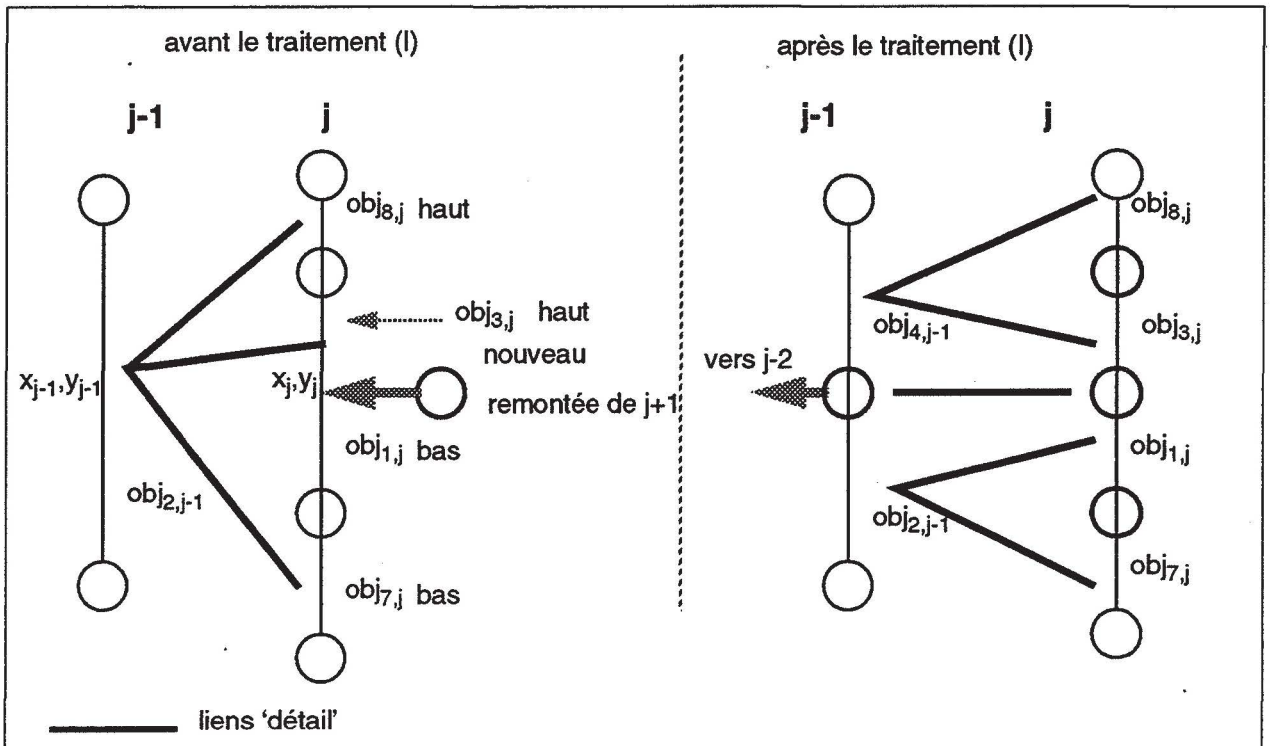


Figure 26 : une étape des mises à jour des liens 'détail' dans les descendants en cas d'ajout d'arête avec éclatement

Ainsi, la mise à jour des liens 'détail' avec les niveaux i+1+h (h > 0) est faite avant le découpage de l'arête *ar* éclatée par l'ajout de ar1. Un traitement du même type que (I) est fait pour mettre à jour les liens 'détail' des objets du niveau i+1 avec obj_i et $obj_{1,i}$ du niveau i. L'objet $obj_{1,i}$ résulte de l'éclatement de obj_i et est associé à une arête possédant un deuxième sommet supérieur lexicographiquement au point d'impact en i. La mise à jour des liens 'détail' avec le niveau parent consiste tout simplement à lier $obj_{1,i}$ par un lien typé 'détail' à l'éventuel unique obj_{i-1} du niveau i-1 anciennement lié à obj_i par un lien détail. L'algorithme s'arrête là puisque la nature sémantique de l'objet obj_{i-1} ne change pas de toute évidence.

5.3. Ajout d'arêtes à une carte planaire ayant une sémantique auxiliaire

De même que pour les arêtes ayant la sémantique de base, les arêtes de sémantique auxiliaire ajoutées qui pourraient nous intéresser dans une mise à jour des liens 'détail' sont celles qui modifient la sémantique d'arêtes pré-existantes ayant des liens 'détail'. Donc, l'ajout d'une arête isolée du reste du dessin d'une sémantique auxiliaire d'un niveau de détail, si cet ajout satisfait les principes de cohérence, ne met pas en cause les liens typés 'détail'. De ce fait, parmi tous les ajouts d'arête acceptables, deux cas seulement nous intéressent :

- un des sommets de l'arête ajoutée est celui d'une arête pré-existante ayant des liens typés 'détail'. Dans ce cas, l'arête ajoutée ne provoque pas d'éclatement d'arête,

- un des sommets de l'arête est strictement à l'intérieur d'une arête pré-existante ayant des liens typés détail. Dans ce cas, l'arête ajoutée provoque un éclatement d'arête.

Contrairement aux arêtes ayant la sémantique de base, les arêtes de sémantique auxiliaire sont clairement définies comme étant la propriété de faces ayant la sémantique de base. L'acceptation de l'ajout d'une arête, préalablement découpée aux frontières des faces détaillées signifie selon le deuxième principe de cohérence (cf. section 2.4.) que cette arête n'appartient pas à une face détaillée.

5.3.1. Ajout d'une arête ne provoquant pas d'éclatement d'arête

Le cas qui nous intéresse est celui de l'ajout d'une arête *ar4* dont un des sommets **Som** est l'un des sommets pré-existants de la carte planaire d'une sémantique auxiliaire. Nous allons successivement traiter le cas où Som est de type différent de **SR2**, puis le cas où Som est de type **SR2** (cf. section 3.2.).

a. Cas où Som est de type différent de SR2

Comme le montre la figure 27, l'ajout d'une arête incidente à un sommet pré-existant Som de type différent de **SR2** (S0 ou S1) n'induit pas la création d'un nouvel objet de type ligne (l_ax). Les liens typés 'détail' des objets de type partie_de_ligne (ax) associés aux arêtes pré-existantes ne sont aucunement affectés. L'existence de Som implique qu'au moins une arête [S0,S1] est incidente en Som. Soient ax1 l'objet partie_de _ligne associé à [S0,S1] et l_ax1 l'objet sémantique ligne qui le contient. L'ajout de *ar4* crée une partie de ligne et modifie la ligne l_ax1 qui est liée éventuellement par un lien typé 'détail' à son homologue dans la carte planaire parente. Ce dernier lien est maintenu, et à partir de ce moment là, la description i pour les sémantiques auxiliaires n'est plus qu'une transformée géométrique de celle du niveau i, mais devient effectivement une description plus détaillée.

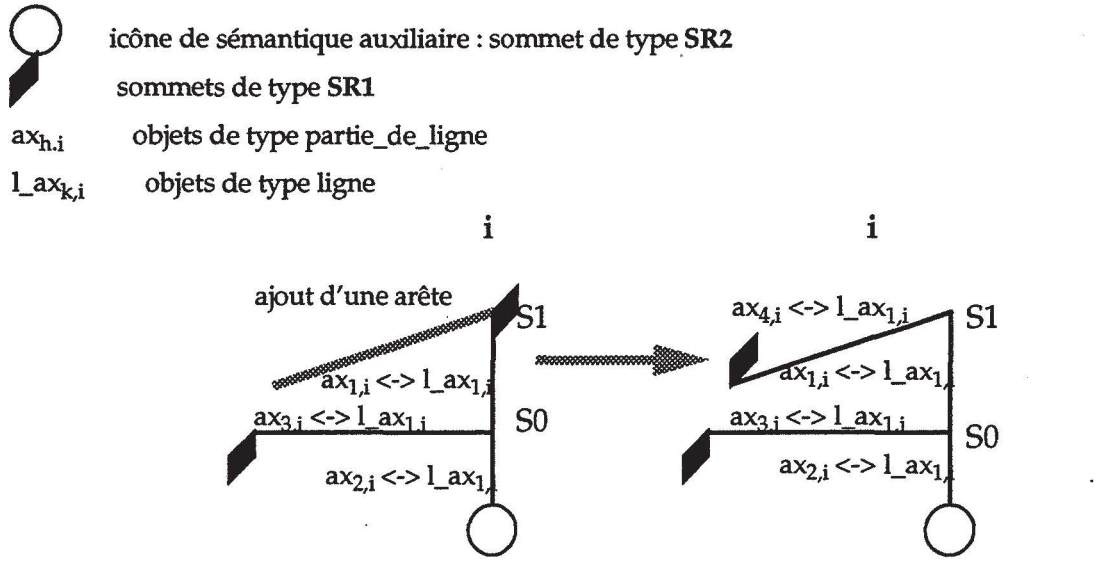


Figure 27 : ajout d'une arête dont un des sommet est confondu avec un sommet existant de type différent de **SR2**

b. Cas où Som est de type SR2

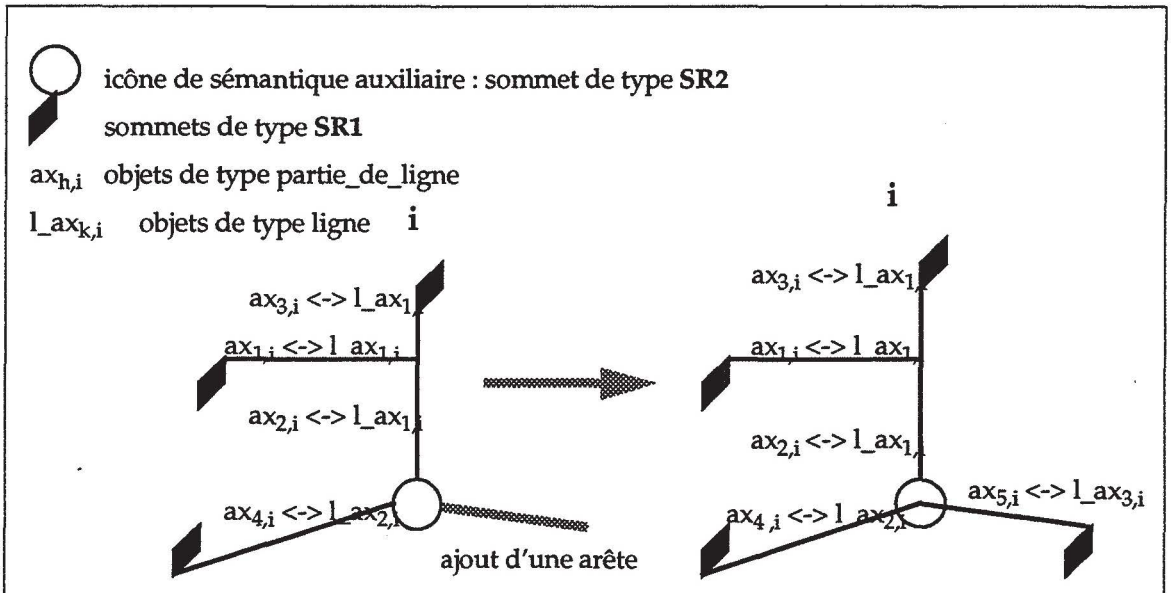


Figure 28 : ajout d'une arête dont un des sommets est confondu avec un sommet existant de type **SR2**

Comme le montre la figure 28, l'ajout d'une arête incidente à un sommet pré-existant Som de type **SR2** induit la création d'un nouvel objet de type ligne. Les liens typés 'détail' des objets de type partie_de_ligne et des objets de type ligne associés aux arêtes pré-existantes ne sont aucunement affectés.

5.3.2. Ajout d'une arête provoquant l'éclatement d'une arête existante

Soient $ax_{1,i}$ l'objet partie_de _ligne associé à l'arête *ar* éclatée et $l\_ax_{1,i}$ l'objet sémantique ligne qui le contient. L'éclatement de *ar* produit un nouvel objet, de type partie_de_ligne $ax_{4,i}$. Comme le montre la figure 29, $ax_{4,i}$ fera partie du même objet ligne $l\_ax_{1,i}$ que $ax_{1,i}$ et n'induit donc pas la création d'un nouvel objet de type ligne. Cependant, un lien typé 'détail' est à mettre en place entre l'objet $ax_{4,i}$ et l'objet éventuellement lié par un lien typé 'détail' à $ax_{1,i}$ dans la carte planaire parente.

L'ajout d'une arête de ce type modifie la ligne l_ax1 qui est liée éventuellement par un lien typé 'détail' à son homologue dans la carte planaire parente. Ce dernier lien est maintenu comme nous l'avons fait dans le paragraphe 5.3.1.a. De ce fait, nous obtenons une description courante plus détaillée que la description parente et la sémantique des liens typés 'détail' ne se limite pas à une simple transformation géométrique.

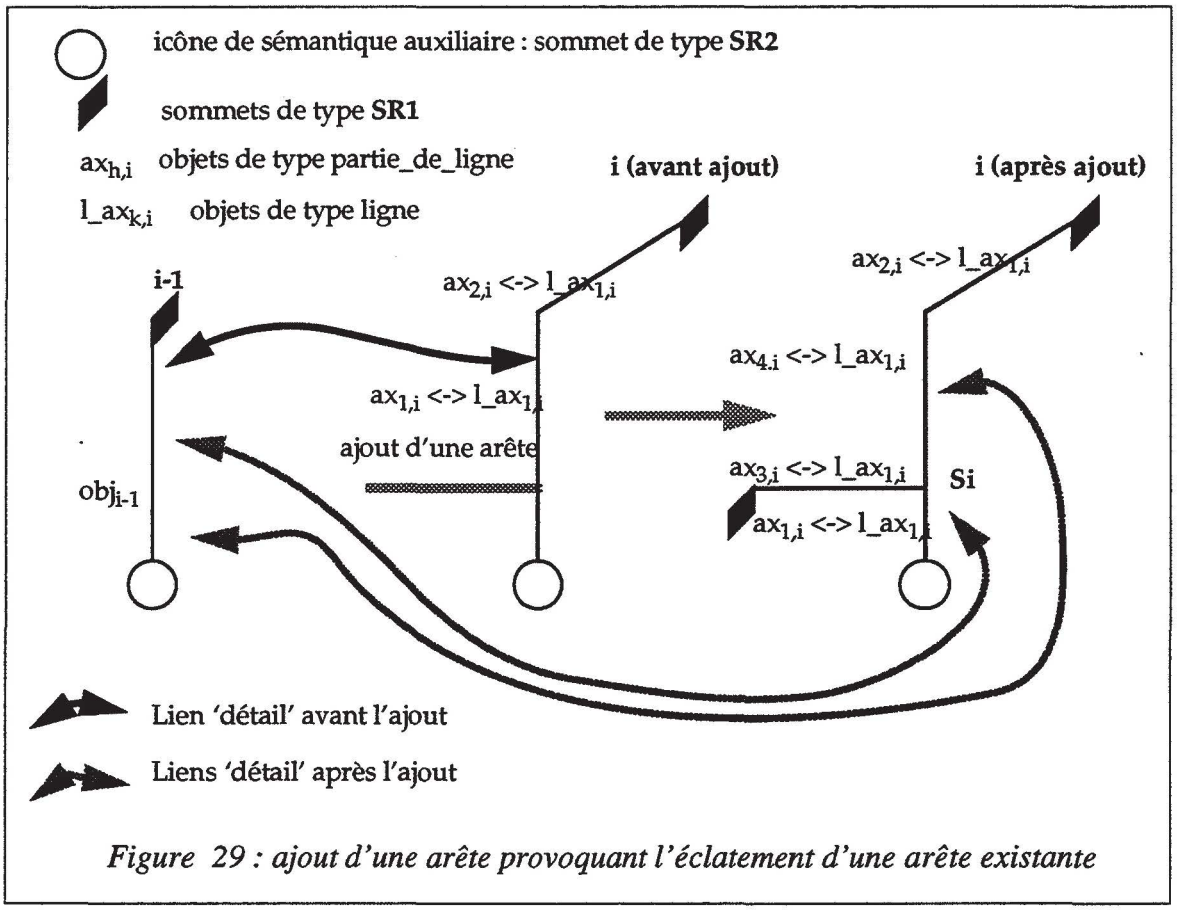


Figure 29 : ajout d'une arête provoquant l'éclatement d'une arête existante

L'algorithme de mise à jour des liens typés 'détail' des objets de type partie_de_ligne, par rapport à ceux de la carte planaire parente, intervient au même endroit que celui du maintien des liens concernant la sémantique de base, quand il s'agit d'éclatement. Cependant, lorsqu'il s'agit d'arêtes de sémantique auxiliaire elle réalise seulement le traitement suivant : découper *ar* associée à $ax_{1,i}$ en **Si** de coordonnées (x,y); ceci aboutit à la création d'un nouvel objet $ax_{4,i}$ de type partie_de_ligne associé à la partie de *ar* au dessus de **Si**. Si une carte planaire parente existe, alors, si un objet obj_{i-1} de cette carte planaire est lié à $ax_{1,i}$ par un lien de type 'détail', il est unique et doit être lié par un lien typé 'détail' à $ax_{4,i}$.

5.4. Ajout d'objets icôniques de connexion XCON à une carte planaire ayant une sémantique auxiliaire

L'ajout d'une icône XCON sur une ligne de sémantique auxiliaire provoque l'éclatement de la ligne au point de l'installation et par conséquent un changement dans les objets sémantiques pré-existants. Nous distinguerons ces ajouts d'icônes selon qu'elles interviennent à un emplacement où existe déjà un sommet (pas d'éclatement d'arêtes), ou bien à l'intérieur strict d'une partie de ligne pré-existante (éclatement d'une arête) dans la carte planaire de sémantique auxiliaire.

5.4.1. Ajout d'icônes de sémantique auxiliaire ne provoquant pas d'éclatement d'arête

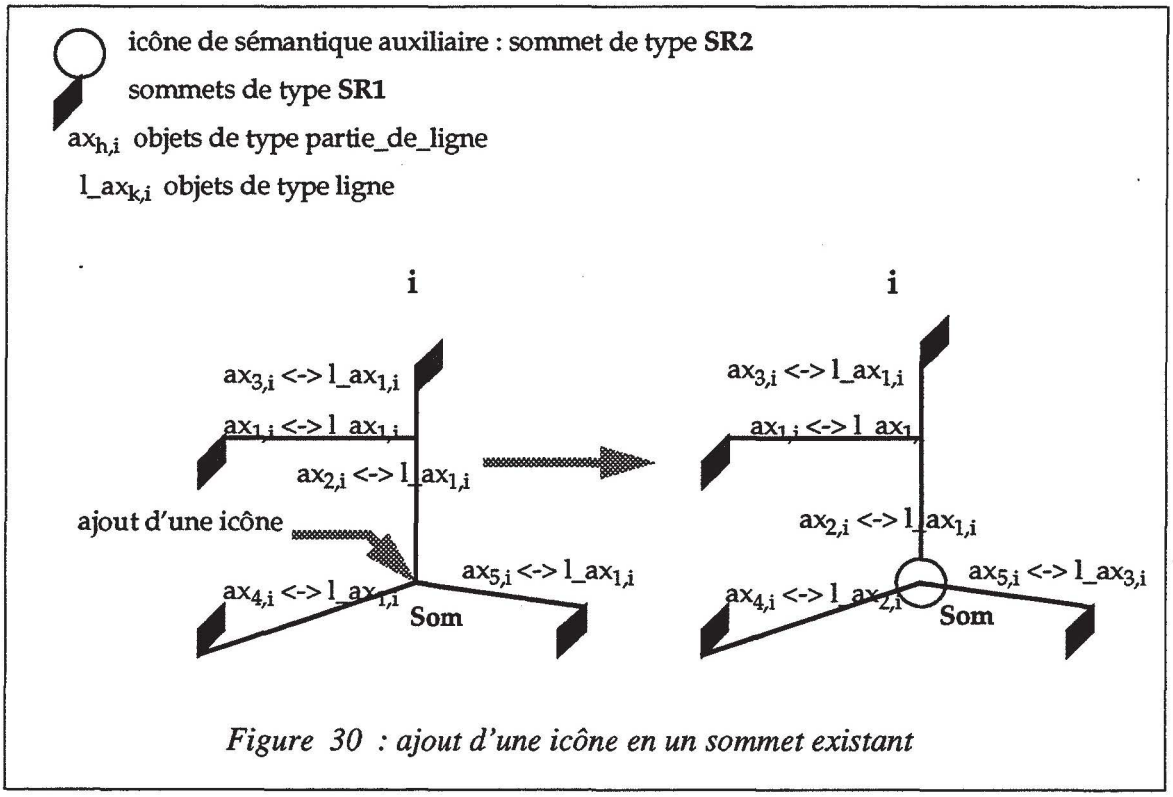


Figure 30 : ajout d'une icône en un sommet existant

Soit CPX la carte planaire représentant la sémantique auxiliaire que possède XCON ajouté au niveau de détail i. L'ajout d'icônes XCON en des sommets de type **SR1** ou **SR2** ne modifie en rien la configuration des lignes et leurs liens typés 'détail'. Nous supposerons donc que l'icône ajoutée est placée en un sommet Som de type différent de **SR1** et **SR2** (cf. figure 30). Les arêtes incidentes à Som sont représentées par des partie_de_ligne axj,i appartenant à la même ligne $l\_ax_{1,i}$. Les liens typés 'détail' des parties de ligne ne sont pas affectés. L'ajout de l'icône crée une ligne différente en chaque arête partant de Som. Si l'ancienne ligne $l\_ax_{1,i}$ possède un lien typé 'détail' avec une ligne l_ax_{-1} d'une carte planaire parente, alors l_ax_{i-1} se retrouve lié par des liens typés 'détail' à toutes les lignes qui ont été créées autour de Som. Ceci est réalisé par le traitement suivant. Dès la création de l'icône, nous la projetons sur l'arête ar2 de CPX qui en est la plus proche. Si le point de projection est proche d'un de ses sommets, alors nous effectuons la superposition du centre de l'icône avec le sommet en question. Soient $ax_{2,i}$ l'objet partie_de_ligne associé à l'arête ar2, et $l\_ax_{1,i}$ la ligne qui le contient. Si $l\_ax_{1,i}$ possède un lien typé 'détail' avec une ligne l_ax_{i-1} de la carte planaire parente, alors affecter la valeur l_ax_{i-1} à la variable globale ZL; sinon ZL pointera sur nil. Dans la phase de vérification des objets de CPX du niveau de détail courant, dès qu'il y a besoin de créer une ligne nouvelle (bien entendu causée par l'ajout de cette icône), nous lions la ligne nouvelle par un lien typé 'détail' à ZL.

5.4.2. Ajout d'une icône de sémantique auxiliaire provoquant l'éclatement d'une arête

Comme le montre la figure 31, l'ajout d'une icône XCON provoquant l'éclatement en Som d'une arête pré-existante *ar* crée une nouvelle arête ar1. Cela induit la création d'un nouvel objet $ax_{3,i}$ de type partie_de_ligne et d'un nouvel objet $l\_ax_{2,i}$ de type ligne contenant $ax_{3,i}$. Soit $ax_{1,i}$, de type partie_de_ligne, anciennement associé à l'arête *ax*. Soit ax_{i-1}, de type partie_de_ligne, tel que : ax_{i-1} fait partie de la carte planaire parente et est lié par un lien typé 'détail' à $ax_{1,i}$. Si ax_{i-1} existe, alors un lien typé 'détail' doit être mis en place entre $ax_{3,i}$ et l'objet ax_{i-1}. Cette mise en place de lien se fait à l'aide d'un traitement similaire à celui décrit dans la section 5.3.2.

Quant aux liens typés 'détail' des objets de type ligne, ils sont mis à jour par un traitement similaire à celui qui a été décrit dans la section 5.4.1.

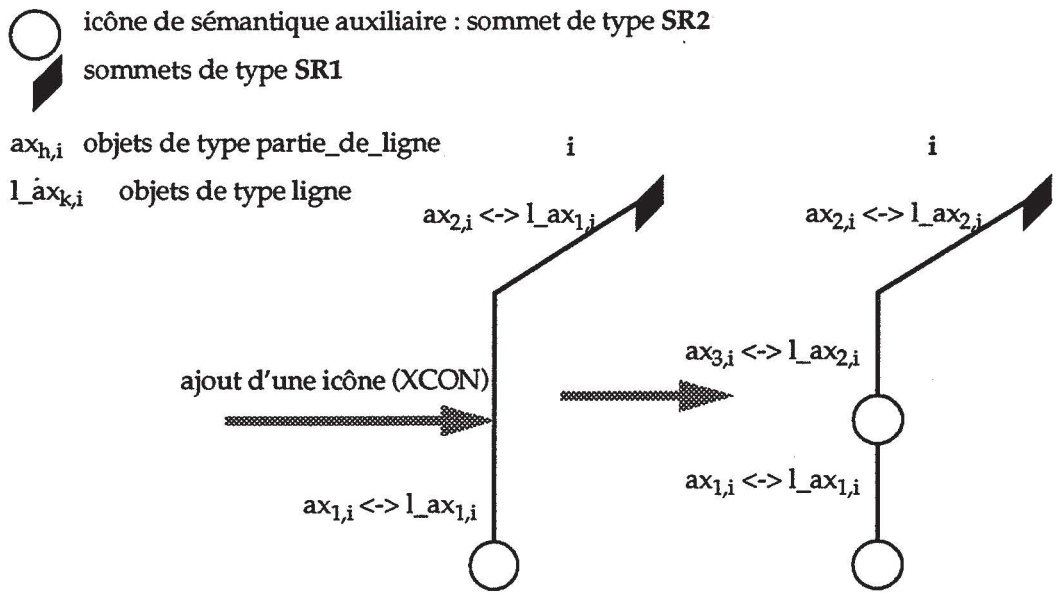


Figure 31 : ajout d'une icône XCON provoquant l'éclatement d'une arête

6. MAINTIEN DES LIENS TYPÉS 'DÉTAIL' EN CAS D'EXPORTATION DE SEGMENTS OU D'ICÔNES VERS UNE CARTE PLANAIRE PARENTE

Nous allons maintenant présenter l'opération d'exportation d'objets en continuant à nous intéresser uniquement aux cas mettant en jeu des liens 'détail'. Le but de l'opération d'exportation est de permettre la remontée d'informations importantes des niveaux de détail plus élevés vers les niveaux de détail moins élevés. Ainsi, l'exportation d'un objet a pour but de créer son homologue direct dans la carte planaire parente dans le cas où cet homologue n'existe pas encore. Or, l'exportation des objets icôniques représentant les éléments de continuité de la sémantique de base ecgsb couvre l'exportation des objets de type pab, et l'exportation d'une ligne de sémantique auxiliaire peut se ramener fonctionnellement à l'exportation de toutes ses parties. Pour les segments, nous traiterons l'exportation des arêtes associées à des objets de type ab (les deux sommets de l'arête ont un sens physique dans la carte planaire courante de base) et des arêtes de sémantique auxiliaire associées à des objets de type partie_de_ligne. Pour les objets icôniques, nous traiterons l'exportation des icônes de connexion XCON pour les sémantiques auxiliaires et des icônes représentant les éléments de continuité de la sémantique de base ecgsb.

Nous commencerons d'abord par l'exportation des objets de type ab; ensuite, nous traiterons l'exportation des objets de type partie_de_ligne; puis, nous finirons cette section par l'exportation des éléments de continuité ecgsb et des icônes auxiliaires XCON.

6.1. Exportation d'un objet ab représenté par une seule arête de la carte planaire de base

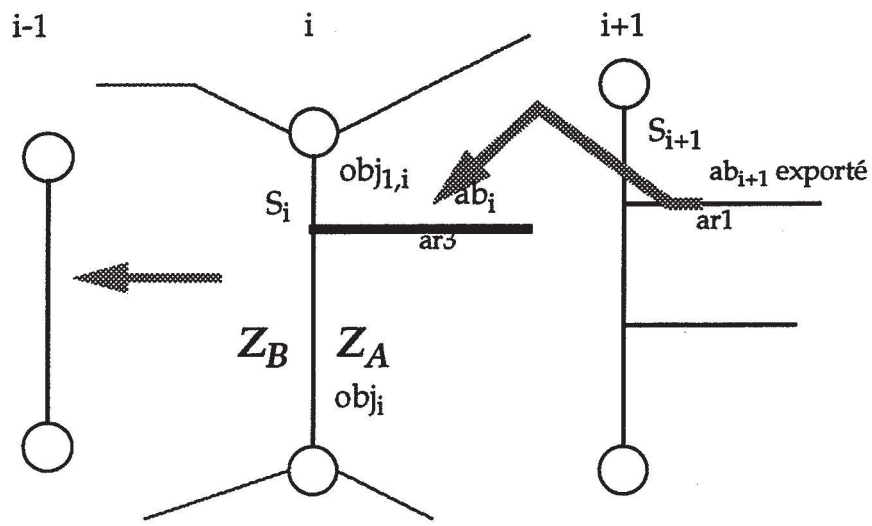


Figure 32 : exportation d'une arête-ab

Soient ar1 l'arête représentant l'objet exporté ab_{i+1} et ar3 l'arête correspondante, par la transformation "Zoom" inverse, dans la carte planaire parente. L'objet ab_i associé à ar3 n'existait pas sinon l'exportation de ab_{i+1} ne serait pas autorisée. Cet objet ab_i est donc créé suite à l'opération d'exportation et est immédiatement lié par un lien typé 'détail' à ab_{i+1}. Deux cas se présentent :

- soit ar3 ne coupe pas les arêtes pré-existantes dans le niveau i , et on s'en tient au traitement précédement décrit,

- soit un des sommets (au moins) de ar3 est le point d'intersection de ar3 avec une (ou deux) arête pré-existante *ar* (cf. figure 32). Dans ce cas, l'objet obj_i représentant *ar* a forcément des liens typés 'détail' avec le fils i+1 qui exporte, comme tous les objets du niveau i contenus dans l'ensemble A des faces à l'origine du niveau de détail courant Z_A.

Dans la suite, nous supposerons que nous sommes dans le second cas avec un point d'intersection S_i autre que l'un des deux sommets de *ar*, ar1 étant non parallèle à *ar* (sinon, il n'y a rien à ajouter à la création de ab_i et à la mise en place d'un lien typé 'détail' entre ab_i et ab_{i+1}).

L'insertion de ar3 dans le niveau parent i passe par l'éclatement de *ar*. Cela transforme obj_i en obj_i et $obj_{1,i}$. Si on suppose que les algorithmes de mise à jour des liens 'détail' respectent, dans leur globalité, le premier principe de la cohérence (cf. section 2.3. de cette partie) et si une carte

planaire parente du niveau i existe, il n'y a qu'un seul objet obj_{i-1} anciennement lié par un lien typé 'détail' à obj_i. La mise à jour des liens 'détail' ne va pas plus loin que le niveau i-1, et pour ce niveau elle consiste à lier par un lien typé 'détail' $obj_{1,i}$ à obj_{i-1}.

Considérons la liste des objets obj_{i+1} de Z_A anciennement liés par des liens typés 'détail' à obj_i. Tous ceux dont la représentation graphique est en dessous de S_{i+1} possèdent des liens 'détail' qui restent valables. Par contre, ceux dont la représentation graphique est au dessus de S_{i+1} voient leurs liens 'détail' avec obj_i s'annuler et se reporter sur $obj_{1,i}$.

Il peut se produire que obj_i ait aussi des liens 'détail' avec des objets obj_{i+1} de détail spécifiés dans un "Zoom" Z_B de faces situées de l'autre côté de *ar* par rapport à A, et dont la représentation graphique fait intervenir *ar*. Dans ce cas, une mise à jour des liens 'détail' de obj_i est à faire par rapport aux objets obj_{i+1} de Z_B. Comme nous le faisons dans le cas d'ajout d'arêtes de base provoquant l'éclatement d'arêtes, la procédure de mise à jour récursive des liens des descendants de Z_B doit être appelée. Ceci créera dans Z_B et ses descendants des éléments de continuité ecgsb et éventuellement des objets de type pab comme l'indique la figure 33. Pour garder une certaine modularité dans nos traitements, nous commençons d'abord par règler le cas des mises à jour récursives de Z_B et, uniquement après, nous mettons à jour les liens 'détail' de $Z_{B,i+1}$ avec les objets de niveau i et les liens 'détail' de $Z_{A,i+1}$ avec les objets de niveau i.

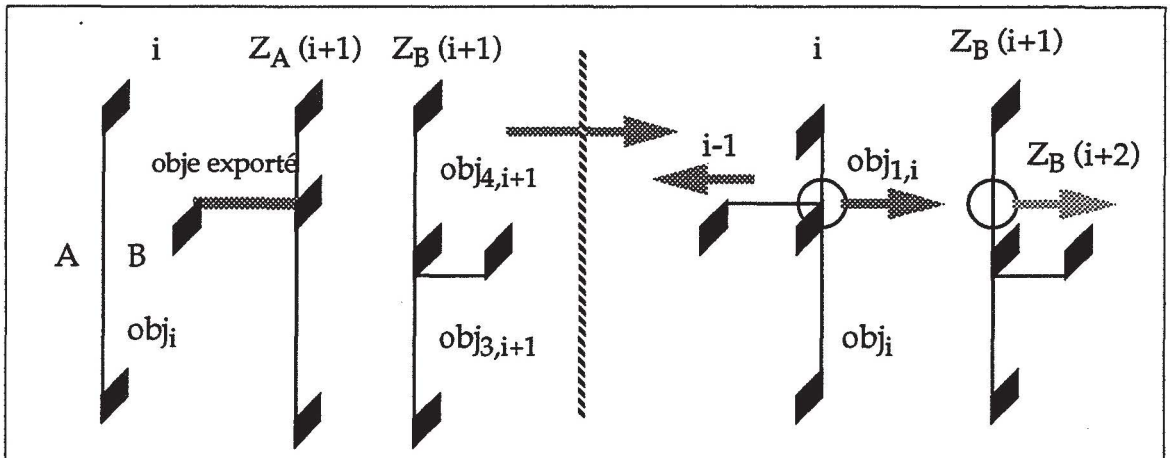


Figure 33 : effet de l'exportation d'une arête-ab sur les objets de niveaux de détail associés à des faces contiguës

Nota bene : l'exportation de plusieurs arêtes d'une carte planaire de base au niveau parent ne doit pas être exagérée. Cela conduirait à la création de nouvelles faces dans le niveau parent par des moyens qui nuisent logiquement à la cohérence de la hiérarchie (cf. section 2.4.).

6.2. Exportation d'une arête de sémantique auxiliaire

Le problème à résoudre est plus simple que celui de l'exportation des arêtes-ab puisqu'il ne peut pas y avoir l'appartenance d'une arête de sémantique auxiliaire à deux faces contiguës. Il faut d'abord créer dans la carte planaire parente un objet de type partie_de_ligne, homologue à obje exporté et établir un lien typé 'détail' entre lui et obje. Puis, dans la phase de vérification des lignes de sémantique auxiliaire de la carte planaire parente, si une nouvelle ligne est créée alors elle est liée par un lien typé 'détail' à celle qui contient obje dans la carte planaire qui exporte.

6.3. Exportation d'un élément de continuité ecgsb

Si l'exportation d'un tel élément obj_{i+1} (du niveau de détail i+1) est acceptée, un élément de continuité homologue obj_i est créé dans le niveau parent et il est lié par un lien typé 'détail' à obj_{i+1} . Le sommet que matérialise obj_{i+1} est le point d'incidence de deux arêtes opposées ayant des liens 'détail' avec la carte planaire parente. Si ces deux arêtes incidentes à obj_{i+1} ont des homologues exacts dans la carte planaire parente, alors le traitement s'arrête et l'état courant des liens 'détail' est globalement cohérent. Supposons donc que ar1, l'une des deux arêtes incidentes à obj_{i+1}, ait un homologue non exact dans la carte planaire parente (après la transformation géométrique "Zoom", l'arête ar3 représentant ar1 dans la carte planaire parente est plus grande que ar1). Soit obj_i l'objet associé à ar3. Une mise à jour correcte de la structure carte planaire et de ses liens 'détail' est résumée dans l'algorithme qui suit.

début

- créer, dans le niveau parent, un nouvel objet $obj_{1,i}$ de type ab correspondant à la demi-arête de ar3 prise par rapport à obj_i dans le même sens que ar1. Soit SOM le nouveau sommet introduit dans le niveau parent i (SOM de coordonnées x et y),

- consulter les objets qui sont liés par des liens typés 'détail' à obj_i et qui appartiennent à la structure LCARTE1=obj_{i+1} ->lcarte; transférer les liens 'détail' des objets dont les arêtes sont dans le même sens que ar1 de obj_i vers $obj_{1,i}$,

- Scruter les liens 'détail' de obj_i avec les objets de niveau i+l. Soit LCARTE2 une structure de données, autre que LCARTE1, dont certaines arêtes possèdent des liens 'détail' avec obj_i. Une des faces de LCARTE2 est alors contiguë à une face de LCARTE1,

- Si LCARTE2 existe

Alors

recursif_lien_descendant(obj_i, i, x, y, objz_cont, LCARTE2),

lier $obj_{1,i}$ par des liens typés 'détail' à ceux du niveau i+1 de LCARTE2, qui étaient liés par un lien 'détail' à obj_i et qui ont été marqués *haut* par recursif_lien_descendant; annuler leur lien avec obj_i. L'annulation des marquages de tous les objets obj_{i+1} de LCARTE2, liés à obj_i, est faite pendant le parcours des liens 'détail' de obj_i; les objets anciennement liés à obj_i et marqués *bas* restent liés à obj_i,

modifier le type des objets obj_i et $obj_{1,i}$ (ab-->pab),

Si une carte planaire parente au niveau i existe

Alors

Si un objet obj_{i-1} de cette carte planaire est lié à obj_i par un lien typé 'détail'

Alors il est unique; le lier par un lien typé 'détail' à $obj_{1,i}$.

fin

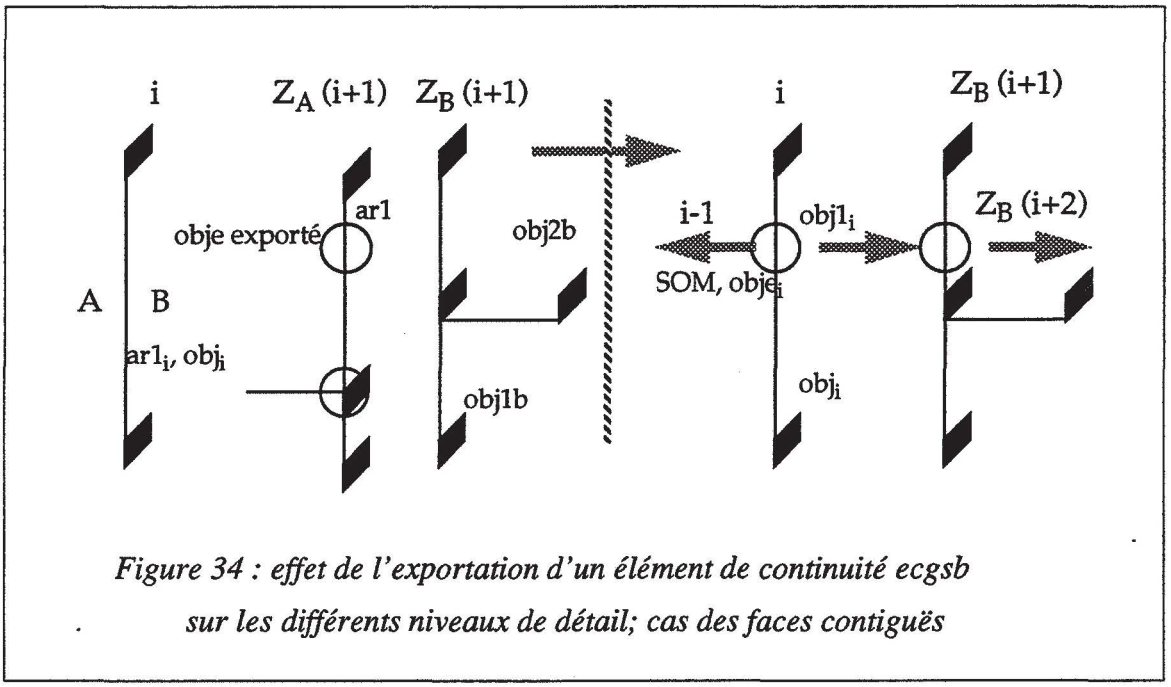


Figure 34 : effet de l'exportation d'un élément de continuité ecgsb sur les différents niveaux de détail; cas des faces contiguës

6.4. Exportation d'une icône de connexion auxiliaire XCON

Nous rappelons que la caractéristique des icônes XCON est que leur ajout sur une ligne provoque son découpage en deux lignes. L'exportation de ces icônes peut provoquer des changements pour certains objets sémantiques et par conséquent pour les liens typés 'détail' de ces objets. Les éléments de continuité géométrique pour les sémantiques auxililiaires en sont un exemple.

L'exportation d'un objet obj_{i+1} de ce type implique avant tout la création d'un objet obj_i qui le représentera dans la carte planaire parente, et la mise en place d'un lien typé 'détail' entre obj_i et obj_{i+1}. Si obj_i se trouve à l'intérieur d'une arête représentant une partie de ligne $ax_{1,i}$ de la ligne $l\_ax_{1,i}$ de la carte planaire parente, alors $l\_ax_{1,i}$ sera coupé en deux lignes $l\_ax_{1,i}$ et $l\_ax_{4,i}$, et $ax_{1,i}$ sera coupé en $ax_{1,i}$ et $ax_{3,i}$. Si $ax_{1,i}$ était lié par un lien typé 'détail' à des parties de ligne $ax_{m,i+1}$ de LCARTE1 égal à obj_{i+1} ->lcarte, alors parmi ces objets nous en avons précisément deux qui sont liés par un lien de sémantique auxiliaire à l'objet obje exporté. Soit $ax_{m2,i+1}$ celui des deux objets qui a son plus petit sommet (pour l'ordre lexicographique) égal au centre de l'objet obj_{i+1}, et $ax_{m1,i+1}$ l'autre objet.

L'idée de la mise à jour des liens 'détail' consiste à utiliser les objets XCON comme intermédiaires pour passer de l'autre côté par rapport à $ax_{m2,i+1}$. Ce qui nous permet sur l'exemple de la figure 35 d'accéder à $ax_{3,i+1}$ et par la suite à la ligne $l\_ax_{3,i+1}$ qui le contient. Ce jeu de bascule peut être contrôlé et limité par un indicateur positionné sur chacun des objets de type partie_de_ligne ou icône dont on ne veut pas se servir. Ainsi dès le départ, nous empêchons l'utilisation de $ax_{1,i+1}$ car tous les liens en dessous de obj_{i+1} dans les niveau i et i+1 restent valides (cf. figure 35). Ce jeu de bascule nous fait arriver sur une partie de ligne $ax_{k,i+1}$, il en existe au moins une qui est $ax_{m2,i+1}$. Nous vérifions si elle a un lien typé 'détail' avec $ax_{k,i}$ de la carte planaire parente. Si $ax_{k,i}$ existe, nous accédons à la ligne $l\_ax_{k,i}$. Si $l\_ax_{k,i}$ est liée par un lien typé 'détail' à la ligne $l\_ax_{k,i+1}$ contenant $ax_{k,i+1}$, et si $l\_ax_{k,i}$ est différent de $l\_ax_{4,i}$, alors nous détruisons le lien entre $l\_ax_{k,i+1}$ et $l\_ax_{k,i}$ et nous lions par un lien typé 'détail' $l\_ax_{k,i+1}$ à $l\_ax_{4,i}$. A la fin de ce traitement, nous positionnons l'indicateur d'arrêt d'utilisation de l'icône XCON courante ainsi que l'indicateur de $ax_{k,i+1}$ et toutes les parties de ligne de $l\_ax_{k,i+1}$, avant d'utiliser $ax_{k,i+1}$ et $l\_ax_{k,i+1}$ pour accéder à une autre icône par l'utilisation de $ax_{k,i+1}$ même ou d'une autre partie de $l\_ax_{k,i+1}$.

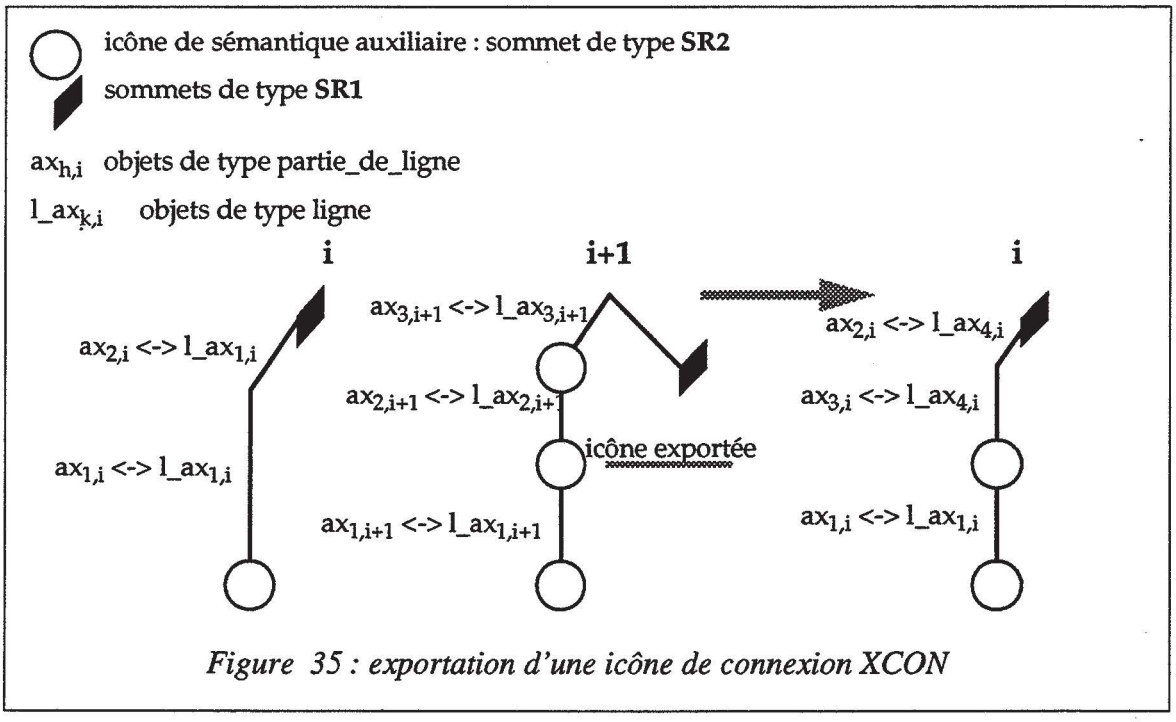


Figure 35 : exportation d'une icône de connexion XCON

7. EXTENSIONS ENVISAGÉES

Dans les sections précédentes de cette partie, nous avons présenté un modèle hiérarchique. Ce modèle a été conçu selon deux principes qui apparaissent d'ailleurs dans les algorithmes d'initialisation : un objet décrit dans le niveau de détail i n'est pas remis en cause et sert comme élément de fondation du niveau de détail i+1. Notre modèle, tel qu'il est décrit par les algorithmes rapportés et testés, ne traite pas le problème de la génération automatique de vues moins détaillées à partir de vues détaillées. En effet, la description d'un niveau de détail i est soit donnée par l'utilisateur aux niveaux i ou i-k, soit complétée par l'utilisateur en utilisant la possibilité d'exporter des objets des niveaux (i+k).

Dans les algorithmes présentant les opérations de base (ajout, exportation), nous avons fait appel à une hypothèse. Cette hypothèse est qu'une arête d'un niveau de détail i est liée par un lien typé 'détail' à une seule arête de niveau i-1 (premier principe). Cette hypothèse est vérifiée par les algorithmes d'initialisation de l'opération "Zoom". Les algorithmes d'ajout et d'exportation sont construits sur cette hypothèse et les traitements qui s'effectuent dans ses algorithmes ne l'altèrent pas. Le modèle hiérarchique basé sur les opérations (initialisation, ajout, exportation) présente une stabilité certaine.

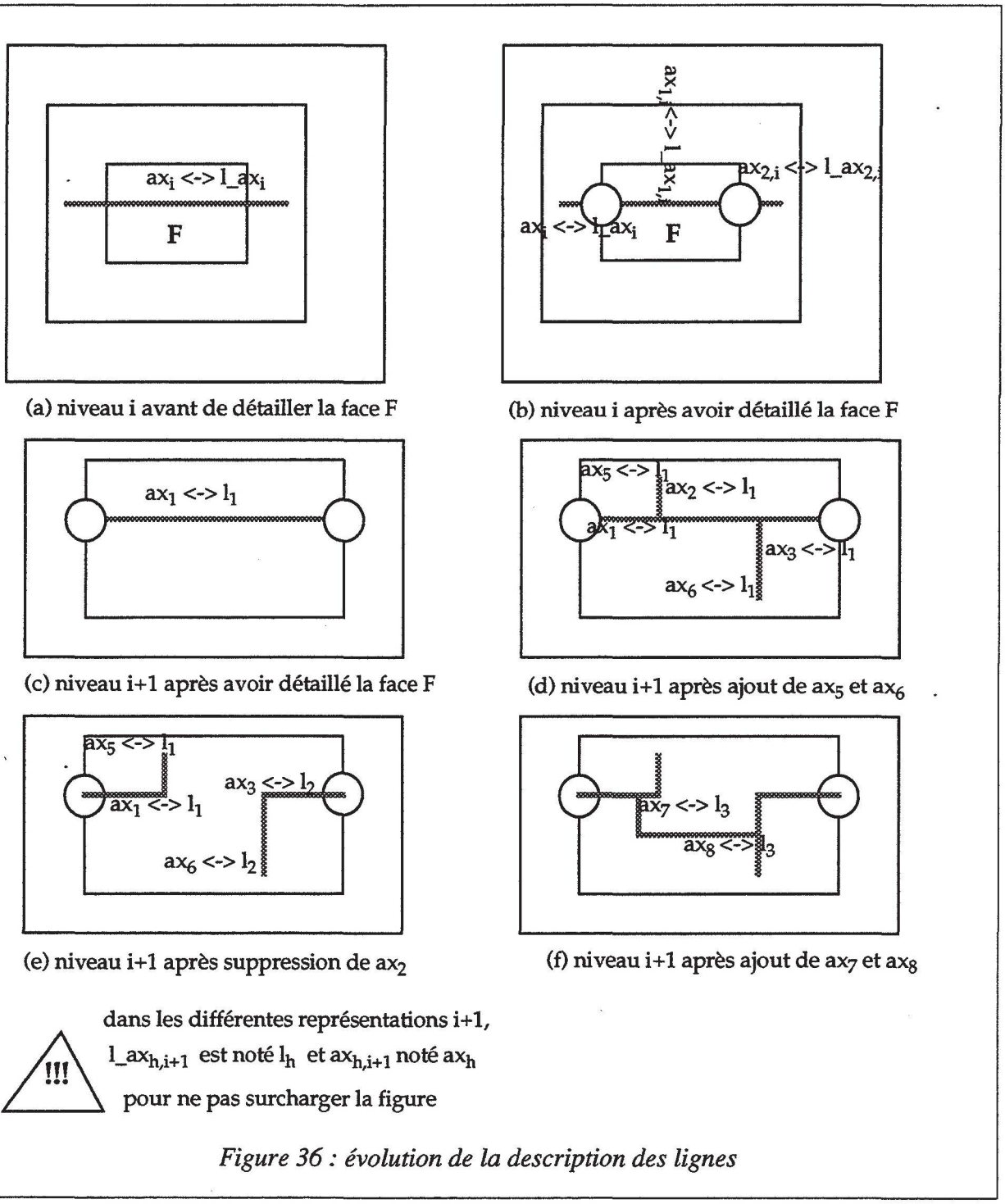
Or, le modèle bi-dimensionnel, présenté dans le chapitre I, offrait la possibilité de supprimer des arêtes d'une carte planaire, et ce de manière incrémentale. Notre souhait pour le futur est de pouvoir concevoir des algorithmes de suppression d'arêtes, ou d'icônes, tout en garantissant la cohérence globale des liens typés 'détail'. De manière plus générale, nous voudrions permettre à un utilisateur d'effectuer des retours arrière dans ses descriptions hiérarchiques de la scène en exploitant au maximum ce qu'il a déjà décrit : fusion entre "Zooms" contiguës, éclatement d'un "Zoom" en "Zooms" contiguës.

Pour le moment, l'utilisateur averti peut quand même faire des suppressions d'arêtes de façon incrémentale sans remettre en cause les liens typés 'détail' ni les deux principes de la hiérarchie. Cela est relativement aisé puisqu'il suffit d'éviter les arêtes délimitant les faces à l'origine du niveau de détail courant et les arêtes ou icônes exportées. Une solution brutale est implémentée pour éviter de redécrire une bonne partie de la scène : une destruction d'une structure Liste_carte ou de l'une de ses cartes planaires est possible sans remettre en cause celles des autres niveaux plans ni la cohérence des liens 'détail' de la Liste_carte parente. La destruction d'une Liste_carte entraine celle de ses descendants hiérarchiques.

Dans les sections qui suivent, nous présenterons d'abord un exemple de suppression d'arête qui n'implique pas de conflit dans l'implémentation actuelle et permet à l'utilisateur d'avoir vraiment une description plus détaillée. Ensuite, nous présenterons le type de détails que l'on espère pouvoir décrire en résolvant les conflits posés par la suppression d'arêtes ou d'icônes ou en introduisant des moyens de description intermédiaires entre le niveau i et ses descendants i+1.

7.1. Cas de la suppression d'une arête

Considérons un exemple simple qui a été testé. Il illustre, d'une certaine manière, comment un utilisateur de notre modèle peut décrire une scène à un certain niveau de détail i et avoir une autre description de cette scène dans le niveau de détail i+1 (cf. figure 36).



(a) niveau i avant de détailler la face F

(b) niveau i après avoir détaillé la face F

(c) niveau i+1 après avoir détaillé la face F

(d) niveau i+1 après ajout de ax_5 et ax_6

(e) niveau i+1 après suppression de ax_2

(f) niveau i+1 après ajout de ax_7 et ax_8

Figure 36 : évolution de la description des lignes

Dans un premier temps, l'utilisateur décrit au niveau de détail i la carte planaire de la sémantique de base sb. Ensuite, il fournit dans ce même niveau de détail la description d'une autre sémantique j (ligne épaisse texturée). Nous supposons qu'une des faces F de la sémantique sb contient des arêtes de la sémantique j et que l'utilisateur souhaite détailler cette face (cf. figure 36(a)). Dans le niveau de détail i+1, l'utilisateur obtient des objets de type ligne (l_j) et partie_de_ligne (ax_j) correctement liés par des liens typés 'détail' à ceux de la carte planaire parente.

Dans le cas de la figure 36(a), l'utilisateur obtient :

- une partie_de_ligne ax_1 liée par un lien typé 'détail' à la partie_de_ligne $ax_{1,i}$ (cf. figures 36(b) et 36(c)),

- ax_1 est lié par un lien 'partie_de' à une ligne l_1,

- l_1 est lié par un lien typé 'détail' à la ligne $l\_ax_{1,i}$ contenant $ax_{1,i}$,

Pour aboutir à la description i+1 de la face F donnée dans la figure 36(f), en gardant des liens typés 'détail' cohérents, l'utilisateur peut entreprendre les actions suivantes :

1- rajouter d'abord les deux parties de ligne ax_5 et ax_6,
(cela produit deux objets partie_de_ligne ax_2 et ax_3 qui sont avec ax_1 liés par des liens 'typés' détail à l'objet $ax_{1,i}$)

2- supprimer la partie de ligne dénomée ax_2,
(cela produit un nouvel objet ligne l_2, non lié par un lien typé 'détail' à la ligne $l\_ax_{1,i}$)

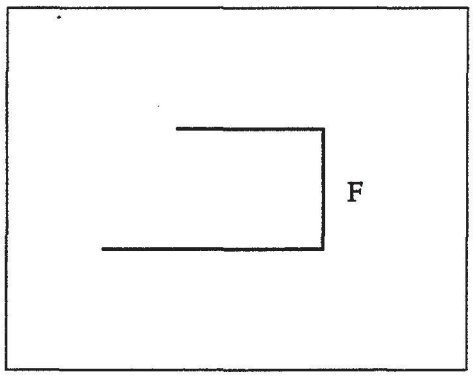
3- rajouter les parties de lignes ax_7 et ax_8 :
(cela produit un nouvel objet ligne l_3, comprenant toutes les parties de ligne du niveau i+1, et lié par un lien typé détail à $l\_ax_{1,i}$: les objets l_1 et l_2 sont donc fusionnés).

7.2. Vers des détails de plus en plus intéressants

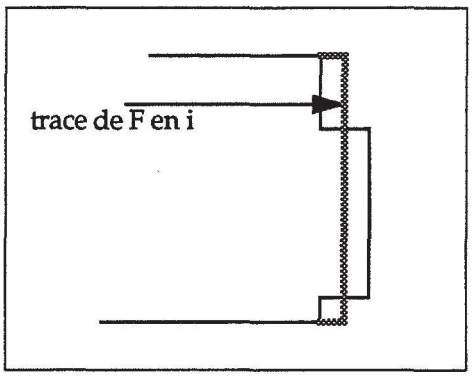
Dans la section 7.1., nous avons vu que, sans développer de nouveaux algorithmes de mise à jour des liens typés 'détail', nous pouvons, en étant averti, construire des descriptions détaillées intéressantes pour des sémantiques autres que celle de base. Pour la sémantique de base, nous pouvons construire des détails intéressants à l'aide de l'opération d'ajout d'arêtes mais uniquement à l'intérieur des faces à l'origine du niveau de détail courant. De nouveaux algorithmes à développer dans le futur devraient nous permettre, à l'aide de la suppression

d'arêtes ou éventuellement d'autres opérations, de mieux décrire les faces à l'origine du niveau de détail courant. Ceci devrait se faire avec une gestion cohérente des liens typés 'détail' et sans remettre en cause les deux principes pour la cohérence de la hiérarchie sur lesquels sont basées les opérations d'ajout et d'exportation.

Les figures 37(a) et 37(b) montrent le type de détail que nous espérons pouvoir saisir pour les faces au niveau de leurs arêtes au contact avec l'extérieur dans tous les niveaux de détail.



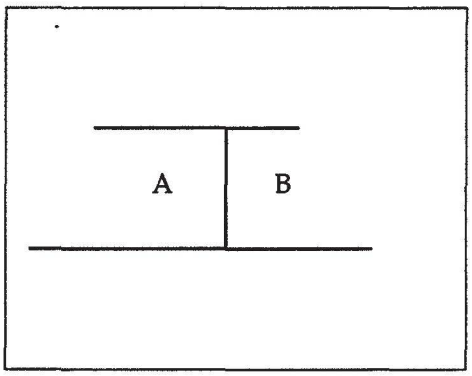
(a) description (i) d'une frontière



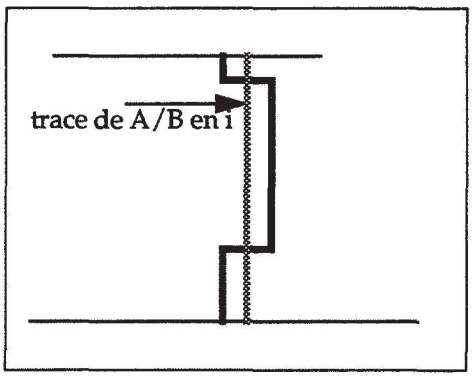
(b) description (i+1) de la frontière F

Figure 37 : description détaillée d'une frontière

Les figures 38(a) et 38(b) montrent le type de détail que nous espérons pouvoir saisir aux niveaux de détail i+1 Z_A et Z_B définis au niveau i par deux faces contiguës A et B.



(a) description (i) de la frontière A/B



(b) description (i+1) de la frontière B

Figure 38 : description détaillée de la frontière de deux faces contiguës

CHAPITRE III

RÉALISATION D'UNE INTERFACE GRAPHIQUE BASÉE SUR LE MODÈLE $2D^{1/2}$ HIÉRARCHIQUE

A. INTRODUCTION

Dans la première partie de cette thèse nous avons décrit une modélisation $2D^{1/2}$ hiérarchique basée sur les cartes planaires. Cette modélisation a été utilisée pour développer une interface graphique connectée à une base de connaissances dans le domaine de la conception et de l'analyse des réseaux informatiques. L'interface graphique devait donc permettre aussi bien à l'utilisateur qu'à la base de connaissances de manipuler, de la manière la plus naturelle possible, les données de réseau ainsi que les données du bâtiment dans lequel le réseau est installé.

D'une manière plus générale, cette base de connaissances a été L'APPLICATION choisie pour valider des travaux de recherche et de développement menés dans le cadre d'un projet européen ESPRIT. L'objectif de ce projet portant le nom MMI^2 est de mettre à contribution les compétences de plusieurs laboratoires européens pour développer une interface multi-modale.

L'organisation de la suite de ce chapitre sera la suivante :

la partie B présentera le projet MMI^2, l'architecture de l'interface MMI^2 et l'ensemble des modes de dialogue offerts par cette interface. Cette partie permettra également de situer l'interface graphique que nous avons développée par rapport aux autres composantes de l'architecture de MMI^2,

la partie C présentera les réalisations effectuées au sein de notre laboratoire et dont la concrétisation est l'interface graphique : outil de manipulation directe de l'interface MMI^2.

B. LE PROJET MMI^2

1. INTRODUCTION

MMI^2 (Multi-Mode Interface for Man-Machine Interaction with knowledge based system) est le projet Esprit II Nº 2474 [BINO 90]. Ce projet a débuté en janvier 1989 pour une durée initiale de trois ans, après lesquels il a été prolongé pour deux années supplémentaires grâce aux résultats prometteurs obtenus au bout des trois premières années. Sept laboratoires de recherche ont contribué à la réalisation de ce projet : BIM (Belgique) le leader du projet, L'Ecole des Mines de Saint-Etienne, l'INRIA (Paris-Rocquencourt), le CRISS (Grenoble), Rutherford Appleton Laboratory (Angleterre), l'Université de Leeds (Ecosse) et ISS (Espagne).

L'objectif principal du projet MMI^2 est de développer une interface homme-machine intégrant plusieurs modes de dialogue : un langage de commande, des langues naturelles (anglais, français et espagnol), le graphique et le gestuel. Les travaux de recherche impliqués par cette interface ont trouvé comme cadre naturel d'application l'interaction avec un système à base de connaissances. En effet, les utilisateurs de ce genre d'application appartiennent à des catégories différentes (techniciens, experts, ingénieurs ...). Les préférences de chacune de ces catégories se forgent pendant la formation, au cours d'une longue expérience dans le domaine de l'application, et enfin par une utilisation soutenue de l'interface elle-même. L'application choisie pour donner corps à nos travaux de recherche a donc été un système expert dans le domaine de la conception et de l'analyse des réseaux informatiques NEST (Network Expert SysTem).

Il était utopique, dans le cadre de ce projet, de penser développer une interface avec une multi-modalité telle que, au sein d'une même action (destruction d'un objet par exemple), l'utilisateur puisse faire concourir plusieurs modes (désignation d'un objet en gestuel ou en graphique, puis indication de l'opération de destruction en langue naturelle ou en langage de commande). Les laboratoires participant à ce projet avaient chacun une compétence dans un des domaines entrant en jeu et avaient, pour la plupart, un prototype démontrant l'état d'avancement de leurs recherches. Par exemple, dans le domaine de la langue naturelle qui est un domaine assez riche, le CRISS disposait déjà d'un prototype permettant l'analyse de la langue française, et un analyseur et générateur de la langue anglaise était déjà en possession de BIM. Concernant la langue espagnole, ISS explorait ses richesses et les mécanismes qui la régissent mais aucun prototype réalisant l'analyse ou la génération n'était développé. L'objectif du projet MMI^2 n'était pas de réaliser, séparément, des analyseurs et générateurs de langue naturelle. L'objectif de MMI^2 était

d'avoir des analyseurs et des générateurs qui puissent coopérer ensemble au sein d'une même application, même s'il est manifeste, au regard de la littérature, que ces langues (français, anglais et espagnol) n'obéissent pas aux mêmes mécanismes.

Un des axes principaux de recherche impliqué par ce projet était donc la définition et la mise en oeuvre d'un formalisme de représentation commun à tous les modes allant des langues naturelles, quelles que soient l'approche et la technique qu'elles utilisent, jusqu'au graphique et au gestuel, en passant par le langage de commande. Une expression correcte à l'aide de ce formalisme doit pouvoir être comprise et exécutée, par chacun des modes, aussi bien en entrée qu'en sortie pour la plupart des modes. Un gestionnaire de dialogue et un planificateur de tâches sont prévus pour décider du mode adéquat à utiliser en sortie et aussi pour prendre l'initiative d'engager des sous-dialogues lorsque des informations manquantes sont à demander à l'utilisateur, en harmonie avec le modèle utilisateur courant.

La portabilité de l'interface pour d'autres bases de connaissances étant un facteur déterminant dans sa réussite, nous avons précisé avec une attention particulière les points d'interconnexion avec l'application, lors de la définition de son architecture.

Dans ce qui suit, nous présenterons d'abord l'architecture générale de l'interface MMI^2. Ensuite, nous présenterons le langage de représentation interne CMR en section 3, et le système expert NEST en section 4. Enfin, nous présenterons en section 5 des extraits du script de validation du premier prototype de MMI^2 qui a fait l'objet d'une démonstration lors de Esprit'91 week à Bruxelles.

2. L'ARCHITECTURE GÉNÉRALE DU SYSTÈME

L'architecture de l'interface a été conçue en utilisant la notion d' "experts" (cf. figure 39). Chaque expert est un module autonome dans le sens où il possède ses propres structures de données. Tous les modules (experts) communiquent entre eux via un formalisme commun de représentation baptisé CMR (pour Common Meaning Representation).

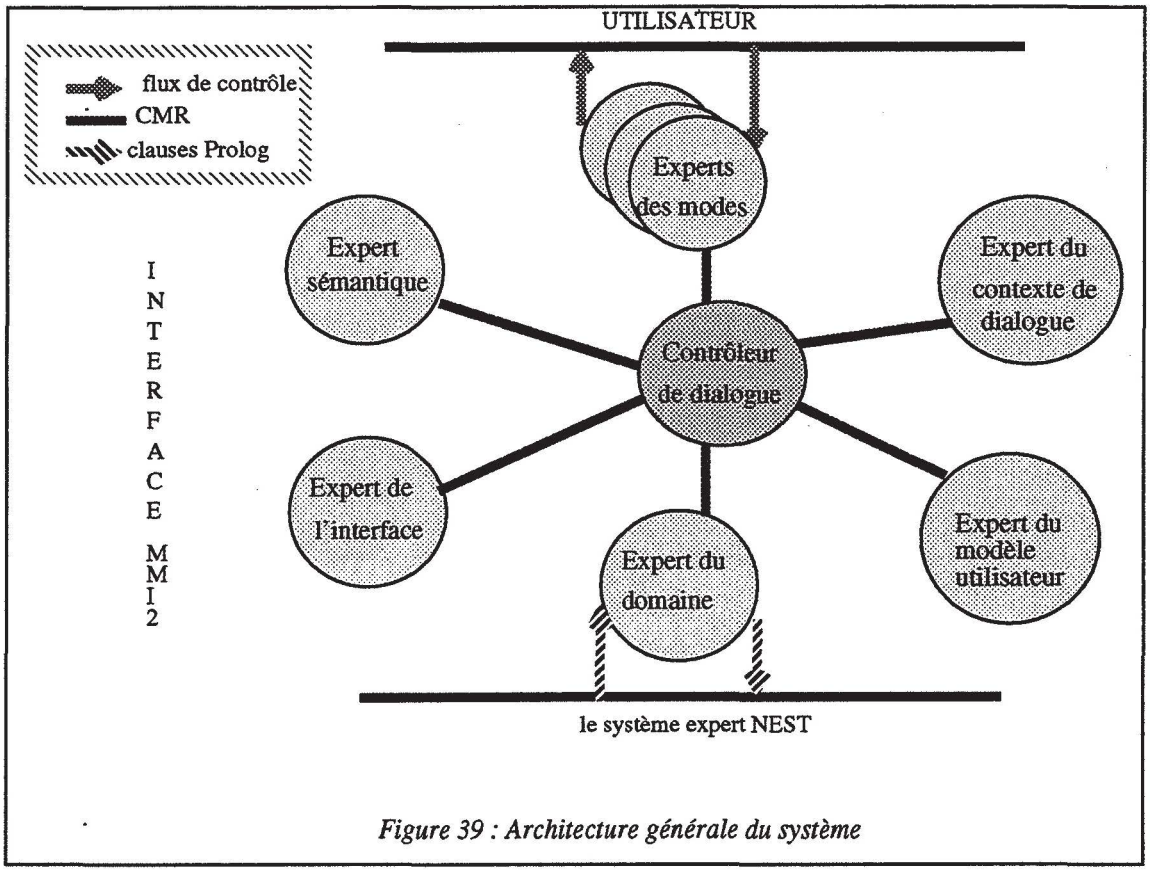


Figure 39 : Architecture générale du système

L'architecture de MMI^2 est du type centralisé : plusieurs experts gravitent autour d'un expert central représentant le contrôleur (ou gestionnaire) de dialogue. Toutes les requêtes passent en général par ce contrôleur après avoir été traduites en CMR. Dans la suite, nous allons présenter ces experts, l'un après l'autre.

2.1. L'expert du domaine

L'expert du domaine possède toute la connaissance relative à l'application. Il dispose de moyens de traduction d'expressions CMR en clauses Prolog, et des moyens de création d'une planification de tâches aboutissant à la résolution d'un problème donné. Cet expert est divisé en deux modules : l'expert formel et l'expert informel.

L'expert formel établit la sémantique formelle des entrées faites par l'utilisateur. Ces entrées lui arrivent sous forme d'expressions CMR. L'évaluation de ces expressions lui permet d'établir le dialogue avec l'application, soit sous forme d'interrogations, soit sous forme de mises à jour.

La raison d'être de l'expert informel est la gestion des buts de l'utilisateur et du système par rapport aux planifications de tâches dont il dispose : dans le cas où les pré-requis d'une planification sont satisfaits, c'est lui qui demande l'exécution des tâches impliquées auprès de l'expert formel; sinon, il fait appel au planificateur des communications pour engager des sous-dialogues dans le but de demander à l'utilisateur d'éventuelles informations manquantes, en prenant en compte le modèle courant de l'utilisateur. Les interactions Nº 24 à 28 du script présenté dans la section 5 permettent de comprendre ce qu'est un sous-dialogue.

2.2. Les experts des modes de dialogue

A chacun des modes de dialogue envisagés dans MMI^2 a été associé un expert. Nous retrouvons donc trois experts associés aux trois langues naturelles qui sont le français, l'anglais et l'espagnol. Vient ensuite un expert associé au langage de commande puis deux experts associés respectivement au graphique et au gestuel. Nous décrivons ci-dessous chacun de ces experts.

2.2.1. L'expert de la langue française

Le module spécialisé dans la langue française est composé d'un analyseur et d'un générateur :

- l'analyseur procède en trois étapes principales. La première étape est l'analyse morphologique du texte entré par l'utilisateur. Elle produit en sortie une liste de séquences ordonnées, de la plus probable à la moins probable. La deuxième étape consiste en l'analyse syntaxique. La troisième et dernière étape effectue l'interprétation sémantico-logique du texte entré sous forme d'expressions CMR. Pour plus d'informations sur l'analyseur nous renvoyons le lecteur aux rapports [DAMI 91] et [EVERT 91].

- le générateur reçoit en entrée des expressions CMR enrichies, par le planificateur de communications, du but de la communication ainsi que du schéma conceptuel du contenu du message véhiculé. Le générateur utilise des connaissances sémantiques (de l'expert sémantique), et des connaissances sur le profil de l'utilisateur et ses préférences (modèle utilisateur), pour générer la proposition qui correspond au mieux à l'expression CMR. Cette génération est effectuée par deux modules appelés respectivement module de planification et module de réalisation [DAMI 93].

2.2.2. L'expert de la langue anglaise

Cet expert intégre un système pré-existant ayant été développé dans le contexte du projet Esprit P107 LOKI [BINO 88]. Le système Loqui dispose de plusieurs modules dont les plus importants sont : un module de morphologie, un module effectuant l'analyse et l'interprétation et servant de corps principal à l'analyseur. Du côté de la génération, on peut distinguer un module

de génération et de détermination des réponses. Tout comme l'analyseur de la langue française, l'analyseur de Loqui a dû être enrichi pour que son lexique englobe des termes spécifiques à l'application. D'ailleurs, cet expert est très lié à l'expert sémantique qui définit un modèle conceptuel du domaine couvert par l'application, en nommant chacune des entités sémantiques du domaine par sa dénomination anglaise, et en la situant par rapport au modèle. Outre l'enrichissement lexical, l'analyseur a dû être modifié pour faire en sorte qu'il produise en sortie des expressions CMR. Cette phase là a été moins évidente pour l'analyseur français (resp. espagnol) qui, au bout de l'analyse, dispose d'une forme sémantico-logique mais avec des termes français (resp. espagnols).

2.2.3. L'expert de la langue espagnole

L'expert de la langue espagnole a été développé au sein de l'ISS (Espagne). Il est composé de cinq modules :

- le module morphologique,

- un module spécial qui enrichit les données par des informations, de nature syntaxique et sémantique, nécessaires à l'analyseur,

- l'analyseur d'espagnol qui effectue l'interprétation sémantico-logique et produit des formules descriptives fonctionnelles.

- le traducteur en CMR, qui transforme les formules descriptives fonctionnelles en expressions CMR,

- le générateur d'espagnol qui permet de présenter à l'utilisateur des questions ou des réponses, formulées par le système, en langue espagnole.

2.2.4. L'expert du langage de commande

Le langage de commande a été prévu pour fournir un moyen simple et efficace pour la réalisation de certaines opérations qui sont, pour la plupart, supportées par d'autres modes. Ainsi, la majorité des opérations peut être formulée en un mot clé, suivi éventuellement par un ou plusieurs arguments. L'expert du langage de commande doit, entre autres, permettre aux utilisateurs expérimentés de programmer des séquences d'opérations qu'ils utilisent fréquemment, en définissant des macro-opérations. La recherche continue de l'efficacité peut aboutir à des contre-performances dues au nombre toujours grandissant des commandes à mémoriser, et à l'impossibilité, pour un utilisateur moyen, d'attribuer des dénominations cohérentes et facilement remémorables. Comme pour les autres modes, des difficultés liées à l'interaction entre modes (déictique, ellipse ...) ont été rencontrées dans le langage de commande. Pour des raisons évidentes de convivialité, ce mode n'est utilisé que dans le sens utilisateur-système.

2.2.5. Les experts graphique et gestuel

Le graphique et le gestuel sont des moyens d'interaction par manipulation directe. Les experts associés utilisent grandement les fonctionnalités supportées par les systèmes de gestion de fenêtres. D'un côté, l'expert graphique est composé d'outils permettant la définition et la manipulation d'objets graphiques. Il est doté de commandes accessibles par manipulation de la souris. Des outils de représentation de données (histogrammes, camemberts ...) font également partie de cet expert. D'un autre côté, l'expert gestuel permet à l'utilisateur de mettre en oeuvre un repertoire de commandes gestuelles et d'en définir de nouvelles. Chaque commande est définie par une forme dessinée (cf. figure 40). Lancer une commande revient à reproduire, à main levée, le geste permettant de dessiner la forme associée.

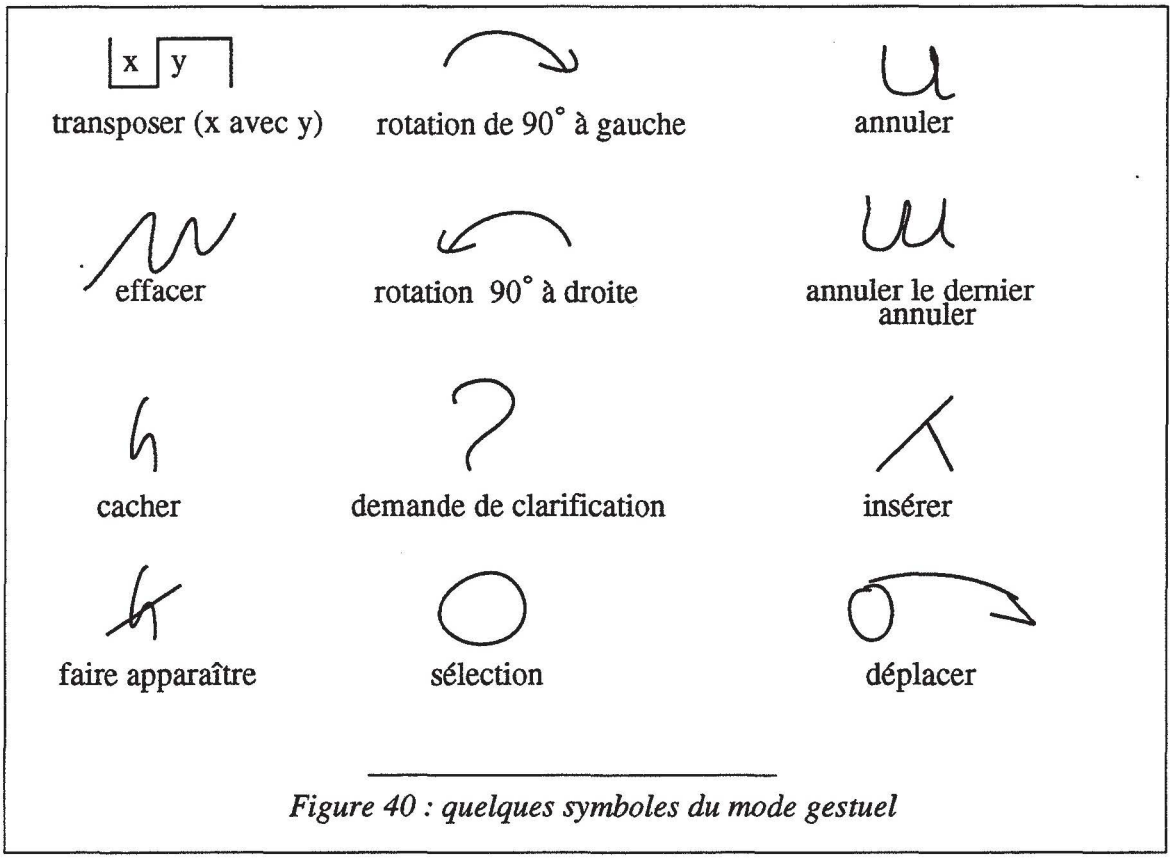


Figure 40 : quelques symboles du mode gestuel

L'expert gestuel travaille en étroite collaboration avec l'expert graphique puisqu'ils utilisent les mêmes fenêtres, et parce que des commandes issues de l'un quelconque des modes, en particulier le graphique et le gestuel, se réfèrent aux mêmes objets graphiques. L'expert gestuel fonctionne uniquement en entrée, tout comme le langage de commande, tandis que l'expert graphique fonctionne en entrée et en sortie : afficher en vidéo inverse un objet sélectionné par une langue naturelle, matérialiser une connexion créée entre deux objets par utilisation du langage de commande ... L'interfaçage entre les outils externes de manipulation et d'interrogation graphique, et le restant des modules de MMI^2 est réalisé par ce que nous appelons le gestionnaire graphique. Ce gestionnaire sert de traducteur entre les outils graphique et gestuel et les structures de données

spécialisées qu'ils manipulent d'une part, et les expressions CMR qui reflètent l'état général des interactions avec l'application d'autre part.

Les experts graphique et gestuel sont représentés ci-dessous au sein d'une même architecture (cf. figure 41). Notre participation au projet se situe au niveau de ce que nous appelons outils de manipulation directe, gestionnaire graphique et bibliothèque graphique. Les développements relatifs à ces composantes sont exposés dans la partie C de ce chapitre.

La principale difficulté de l'expert gestuel réside dans la reconnaissance des symboles dessinés par l'utilisateur. L'algorithme qui l'effectue s'appuie sur un arbre de décision, et en cas d'échec, un menu contenant tous les symboles est affiché sur l'écran, permettant ainsi à l'utilisateur de choisir sans ambiguïté le symbole désiré.

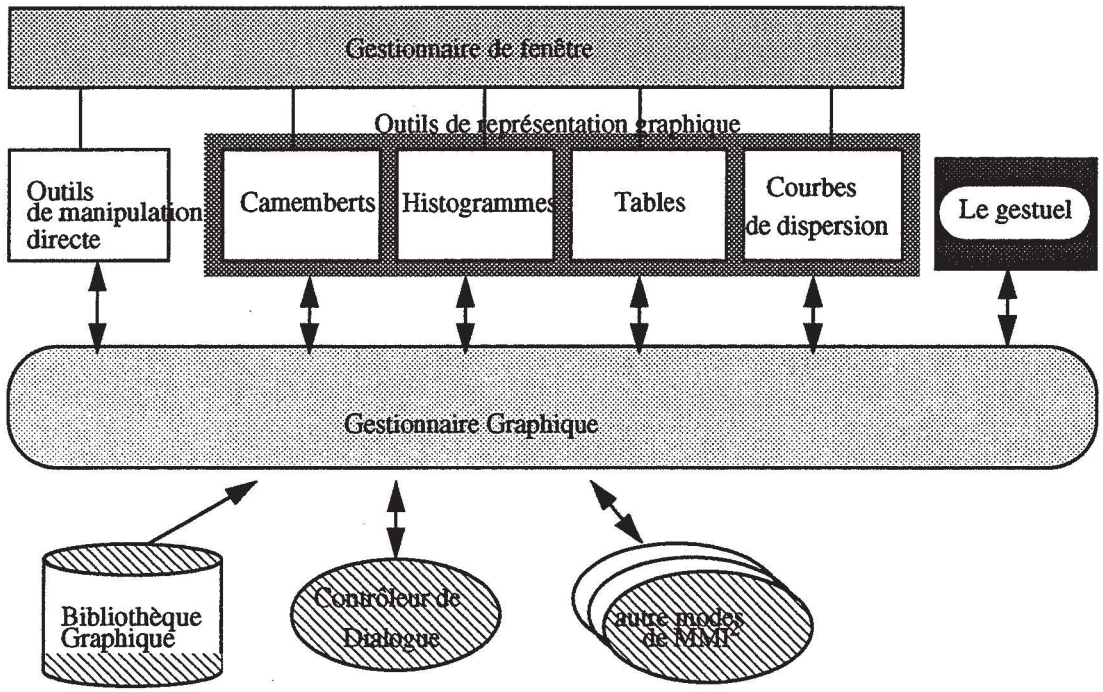


Figure 41 : Architecture incluant les experts des modes graphique et gestuel

2.3. L'expert sémantique

Le rôle de l'expert sémantique est de définir un modèle conceptuel du domaine couvert par l'application, en nommant de façon appropriée chacune des entités sémantiques du domaine, et en la situant par rapport au modèle. De plus, cet expert doit procurer à l'interface la possibilité de s'auto-décrire. Sa conception a été guidée par les nécessités suivantes :

- l'expert sémantique devait offrir la terminologie nécessaire à la formulation d'expressions CMR communes à tous les modes, y compris les différentes langues naturelles et le graphique,

- l'expert sémantique devait profiter de l'expérience du projet Esprit I ACORD [KRIS 88]. En effet, ce projet intégrait le graphique et la langue naturelle pour l'interrogation et la mise à jour d'une base de connaissances. Son langage de représentation sémantique InL a montré que l'interprétation des objets graphiques comme des noms était inadaptée. Notre expert sémantique se devait d'explorer une autre éventualité : ne pas modéliser les objets graphiques en tant que tels, mais modéliser les interactions graphiques.

L'expert sémantique a donc été défini sur la base de ce que nous appelons un type sémantique. Ce type peut être lié soit à des objets (personne, machine ...), soit à des actions (création, déplacement ...) soit à des situations (être connecté à ...). L'expert sémantique associe aux types sémantiques des rôles ou des propriétés qui sont utiles pour interpréter les entrées de l'utilisateur et détecter les incohérences et les formulations inadaptées sans avoir à accéder à l'application.

2.4. L'expert du modèle utilisateur

Cet expert contient des informations concernant les utilisateurs. Ces informations peuvent être utilisées par les différents modules du système. Le modèle choisi est un modèle orienté objet. A chaque classe d'utilisateurs est associée une description générale des connaissances qu'est censé posséder chaque membre de la classe. De plus, chaque utilisateur possède une description individuelle qui contient des informations qui lui sont propres, comme ses préférences, ses capacités et ses croyances. L'implémentation du modèle utilisateur est inspirée de GUMS (General User Modeling Shell) [FINI 89].

Des expériences du type magicien d'Oz (simulation d'un système par un expert humain) ont été conduites auprès des utilisateurs potentiels de la future application. La seule différence avec le magicien d'Oz est que, malgré le mur qui sépare les pièces où se trouvent la machine de l'utilisateur d'une part, et la machine de l'expert d'autre part, les utilisateurs savent qu'ils dialoguent avec un expert humain et non avec un système expert. Ces expériences ont eu un double intérêt :

- formuler un ensemble de règles qui, à partir des interactions avec un utilisateur, permettent de statuer sur son appartenance à une classe particulière et d'adopter ensuite une règle de conduite avec lui (plus ou moins de prise d'initiative par le système, plus ou moins d'étapes intermédiaires dans la présentation des résultats ...),

- extraire un sous-langage de la langue naturelle spécifique au domaine de l'application, car quoi qu'on fasse et malgré les progrès technologiques, plus le langage à analyser par ordinateur est riche, plus les temps de réponse sont des obstacles à l'interaction.

Dans cette expérience, les utilisateurs appartenaient à quatre catégories :

- des technico-commerciaux spécialistes de la conception des réseaux au niveau physique (la topologie et les composants matériels du réseau),

- des spécialistes des logiciels de réseau qui interviennent pour résoudre des problèmes de compatibilité entre les différents composants matériels du réseau,

- des technico-commerciaux spécialistes de vente de matériels informatiques sans être pour autant spécialistes des réseaux,

- des ingénieurs informaticiens qui ne sont pas des spécialistes des réseaux mais qui ont différents niveaux de connaissance dans ce domaine.

L'expert humain est un expert des réseaux qui a une très bonne connaissance dans le domaine de la conception des réseaux informatiques et occupe un poste de conseiller auprès des technico-commerciaux.

2.5. Le contrôleur du dialogue

S'agissant de l'utilisation d'une application (base de connaissances) via une interface, un contrôleur de dialogue est incontournable pour les raisons suivantes :

- un utilisateur peut changer de but à tout moment de l'interaction,

- une incompréhension ou le fait que l'utilisateur envisage un type de solution au problème qu'il se pose, peut le conduire à répondre à une question par une autre question,

- des réponses ambiguës ou incomplètes de l'utilisateur doivent entrainer l'enclenchement d'un mécanisme appelé sous-dialogue.

De nombreuses recherches portent sur ce genre de problèmes et notamment sur le niveau où doivent être placés les moyens de décision (interface ou application) concernant la supervision du dialogue. Comme on peut le constater sur le dessin de l'architecture générale, nous avons opté pour un dispositif de décision placé au coeur de l'interface, que nous avons baptisé contrôleur de dialogue.

Le dispositif de décision repose sur le contexte du dialogue, qui est une pile que l'on incrémente ou décrémente par exécution des décisions. Cette pile contient un ensemble d'attitudes provenant du système ou de l'utilisateur. A une question de l'utilisateur est associée l'attitude Uwk

(the user wants to know), à une commande de l'utilisateur est associée l'attitude Uw, et, enfin, à une assertion est associée l'attitude UwSk (S pour le système).

Le dispositif de décision peut trancher par son accès à des structures de données intitulées "*plan de tâche*", "*plan de communication*" et "*espace de référence de discours*". Cette dernière donnée porte également le nom de contexte de dialogue.

2.6. L'expert du contexte de dialogue

Le contexte de dialogue porte également la dénomination d'espace de référence de discours. Chaque objet mentionné dans une interaction (venant de l'utilisateur ou du système) doit exister dans le monde de discours où il est appelé référence de discours. Actuellement, cet expert est utilisé pour la résolution des anaphores et des ellipses. Quand une expression CMR n'est pas résolue, c'est à dire qu'elle contient une ou plusieurs anaphores ou ellipses, elle est transmise au contrôleur de dialogue qui fait appel à l'expert du contexte de dialogue. En effet, ce dernier dispose de l'information contextuelle nécessaire à la résolution de l'expression CMR.

L'idée de base pour l'intégration des modes est de permettre la désignation d'un ensemble appelé *espace de référence de discours* commun à tous les modes. Chaque mode possède ses propres mécanismes pour la création, la manipulation et la désignation des *références de discours*. Cependant, l'interface doit maintenir une représentation unique de la référence, même s'il y a plusieurs descriptions dans différents modes. Le rôle de l'expert du contexte de dialogue est d'identifier les structures de dialogue et ses différentes composantes, d'enregistrer ces structures et d'en extraire les informations utiles.

Un exemple d'anaphores envisagées par l'application, et résolues par l'expert du contexte de dialogue, figure dans l'interaction N^{o} 5 du script décrit dans la section 5 de ce chapitre.

2.7. L'expert de l'interface

Dans MMI^2, les informations déclaratives concernant l'interface font partie des connaissances de l'expert de l'interface. Ces informations sont consultées par le contrôleur de dialogue chaque fois que l'utilisateur demande des précisions sur les limites, les capacités, les structures et les composants de l'interface. Parmi les rôles de cet expert figurent :

- la transmission des réponses du système au mode concerné,

- l'évaluation formelle des expressions CMR qui ne relèvent pas de la compétence de l'application,

- la gestion des fenêtres associées aux différents modes.

3. LE LANGAGE DE REPRÉSENTATION INTERNE : CMR

CMR (pour Common Meaning Representation) est un langage permettant de décrire des interactions exprimées dans l'un quelconque des modes de communication de l'interface. Pour cette raison, nous parlons en termes d'actes de communication. Chaque acte de communication doit véhiculer un référent de discours (étiquette attribuée par l'expert sémantique et compréhensible par l'ensemble des modes). Un acte de communication contient une forme propositionnelle, une forme logique, une force élocutionnaire et des annotations dont la principale source est la langue naturelle. Ainsi, la représentation CMR doit permettre d'identifier les objets, leurs propriétés et les relations qui existent entre eux. Elle doit en plus être dotée d'une structure de données permettant d'inclure des informations additionnelles du type : mode de communication, présuppositions de l'utilisateur ...

Le consortium du projet a décidé de prendre comme base de la CMR la logique du premier ordre avec quelques extensions (le quantificateur "The" permettant d'utiliser des termes génériques, les quantificateurs numériques : exactement-3, moins-de-3, au-plus-3 ...). Pour plus d'informations sur ce langage, le lecteur pourra se reporter à [BIM 89].

4. LE SYSTEME EXPERT NEST

NEST est un système à base de connaissances pour la configuration de réseaux locaux implémentant les technologies Ethernet et TCP/IP. Dans NEST, on trouve une base de données commune, un outil d'analyse et un outil de conception de réseaux.

La base de données commune est implémentée sous forme d'une hiérarchie de classes. Compte tenu de l'application, nous retrouvons au sommet de la pyramide des classes, une classe générique intitulée "Network_parts" et une autre classe générique intitulée "Building_structures". L'utilisation d'une couche orientée objet BIM_probe, au dessus de Prolog_by_BIM, couplée à un langage de programmation par contraintes (PCL), a permis de définir la hiérarchie des classes. Les objets composant la base de données ont pu être définis sous forme de hiérarchies "ISA" avec héritage multiple, spécialisation d'attributs, attachement procédural (méthodes) ...

L'outil d'analyse de réseaux effectue plusieurs types d'analyses. Pour un réseau donné, l'outil peut effectuer, par rapport aux connaissances dont il dispose sur les technologies Ethernet, une analyse de validité, une analyse d'extensibilité, une analyse de relation client-serveur, une analyse de départementalisation, et enfin une analyse du coût.

L'outil de conception est complètement automatique. Il se base sur une étude faite auprès des experts humains dans le domaine [DARS 90]. Cette étude a révélé que face à un problème de conception arbitré par des contraintes de coût, des contraintes sur le bâtiment (ancien, rénové ou pas encore construit) ... les experts humains adoptent l'approche classique de décomposition d'un problème en sous-problèmes. L'utilisateur décrit la topographie du bâtiment, ses besoins et les contraintes (cf. toutes les interactions du script rapportées dans la section qui suit). L'outil de conception en déduit une structure globale pour le réseau, le câblage de chaque étage, le choix des câbles ...

Pour plus d'informations sur NEST, le lecteur peut se reporter à [JOUV 92].

5. EXTRAITS DU SCRIPT DE VALIDATION DU PREMIER PROTOTYPE

Le format général des interactions avec le système est représenté comme suit : <numero acte> <agent> <mode> <contenu>.

Les abréviations suivantes ont été adoptées : CL pour Command Language, EM pour English Mode, GR pour Graphics. En ce qui concerne l'agent, nous avons adopté u pour désigner l'utilisateur et s pour désigner le système. Le script a été rédigé et testé en utilisant principalement la langue anglaise, projet Esprit oblige. De ce fait, nous ne trouverons pas dans les extraits présentés des traces des modes de dialogue en langue naturelle française ou espagnole. Nous tenons à préciser que des scripts français et espagnol ont été rédigés et testés.

Les interactions suivantes sont relatives aux descriptions fournies par l'utilisateur lors de la conception d'un réseau informatique :

(1) u (CL) : name mdw

(2) s (EM) : ok

(NOTA : le système répond en utilisant la langue anglaise, d'une part parce que la langue naturelle a été jugée le moyen adéquat, convivial, pour commencer le dialogue, et d'autre part parce que l'utilisateur mdw a été catalogué comme utilisateur préférant la langue anglaise)

(3) u (EM) : Put 2 X_Terminals on a network.

(4) s (EM) : ok

(5) u (EM) : The network should have 4 Sparc_SLCs.

(6) s (EM) : ok

(7) u (EM) : Put the network in a building.

(8) s (GR) : puts up tool

(Le système, via le contrôleur de dialogue, envoie une requête à l'expert graphique. Cette requête CMR est traduite dans le gestionnaire graphique en prédicat permettant l'affichage, au premier plan, de l'outil d'interaction graphique développé dans notre laboratoire et permettant de décrire la topographie du bâtiment dont a besoin NEST pour effectuer la tâche de conception)

(9) s (EM) : what are the specifications of the building ?

(10) u (GR) : 'draws the building and activates the button done when finished'.

(11) s (EM) : what is the budget for the network ?

...

...

(20) u (EM) : Add a SUN3 to the network.

(21) s (EM) : what is the location of the cmr_SUN3_98 ?

(22) u (GR) : selects Room1.

(23) s (EM) : ok <ajoute un SUN3 au réseau et l'affiche à l'écran>

(24) u (EM) : Add a workstation to the network.

(25) s (EM) : Sorry.

(26) s (EM) : The location and type of the workstation cmr_WORKSTATION_99 are underspecified.

(27) s (EM) : Adding a workstation requires specification of type and location.

(28) s (EM) : What is the type of the cmr_WORKSTATION_99 ?

...

...

Dans la figure 42 nous avons reproduit une copie écran de l'interface MMI^2 pendant l'interaction avec le système à base de connaissances NEST.

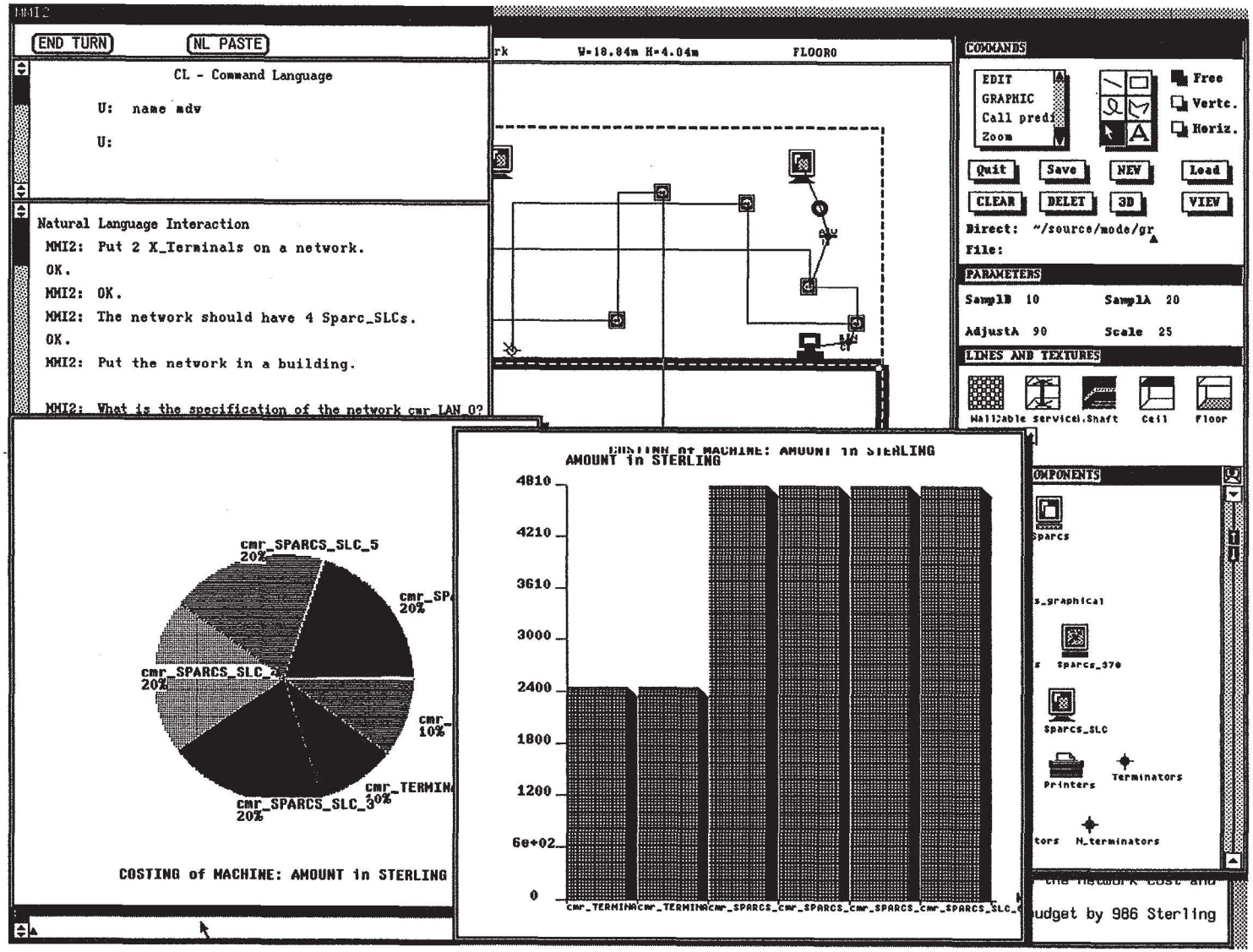


Figure 42 : MMI^2 : une vue globale

C. LA RÉALISATION

1. INTRODUCTION

La réalisation, à laquelle nous consacrons cette partie, se limite aux travaux effectués dans notre laboratoire et dont le but est l'élaboration de l'expert graphique. Comme nous l'avons dit dans la partie précédente relative au projet MMI^2, l'expert graphique constitue une composante à part entière dans l'architecture générale de l'interface multi-modale MMI^2. Une présentation d'une architecture regroupant l'expert graphique et l'expert gestuel a déjà été faite dans ce chapitre (cf. figure 41 de la partie B). Les laboratoires ayant contribué à la définition de cette architecture et aux développements sous-jacents sont : Rutherford Appleton Laboratory (Angleterre), l'université de Leeds (Ecosse) et le département d'informatique appliquée de l'école des Mines de Saint-Eienne. L'architecture regroupant les experts graphique et gestuel est principalement constituée de deux niveaux : un niveau externe de représentation manipulable par l'utilisateur, et un niveau interne de représentation qui est plutôt tourné vers les autres modes de MMI^2.

La partie du niveau externe de représentation lié au mode gestuel a été réalisée au sein du laboratoire de Leeds. Les notions de base relatives au mode gestuel ont été présentées dans la partie B de ce chapitre. Pour plus d'informations sur ce sujet, nous conseillons au lecteur de se reporter à [HOWE 91].

La partie du niveau externe de représentation lié aux outils de représentation graphique (camemberts, histogrammes, tables, courbes de dispersion) a été réalisée au sein du laboratoire de RAL (Rutherford Appleton Laboratory). Les travaux relatifs à cette partie sont exposés dans [CHAP 91]. La copie d'écran, que représente la figure 42 de la partie B, faite pendant une interaction avec l'interface MMI^2 montre l'aspect externe de ces outils.

La dernière partie du niveau externe de représentation concerne l'outil de manipulation directe (OMD). L'OMD regroupe un certain nombre d'outils permettant de représenter les objets de l'application, en termes géométriques, topologiques et sémantiques (topographie du bâtiment, longueur des câbles, adjacence des pièces ...). Il a été développé au sein de notre laboratoire [BENA 92b].

En ce qui concerne le niveau interne de représentation, regroupant le gestionnaire graphique et la bibliothèque graphique, les travaux ayant permis sa conception et son développement ont été effectués en collaboration entre les trois laboratoires. Ce travail de

collaboration s'accentuant lors de l'intégration de l'ensemble des modes au sein d'un même système : l'interface MMI^2 connectée à l'application NEST (Network Expert SysTem).

Ce qui suit est consacré en grande partie à l'outil de manipulation directe que nous allons présenter en section 2. La section 3 explique les deux modes de fonctionnement entrée et sortie du mode graphique et fournit un exemple montrant le flux d'information entre les composants de l'interface MMI^2.

2. L'OUTIL DE MANIPULATION DIRECTE (OMD) - LIEN AVEC LE MODÈLE $2D^{1/2}$ HIÉRARCHIQUE BASÉ SUR LES CARTES PLANAIRES

L'application choisie pour valider l'interface MMI^2 est un système expert dans le domaine de la conception et de l'analyse des réseaux informatiques. Pour effectuer une analyse de réseau, l'application doit connaître les éléments qui le composent, leurs emplacements ... Comme le lecteur peut le constater dans les extraits du script de validation de l'interface MMI^2, script tourné vers la conception, l'application demande à l'utilisateur de lui fournir la topographie du bâtiment dans lequel le réseau doit être installé (se référer à la section B de ce chapitre).

Les objectifs assignés à l'outil de manipulation directe développé au sein de l'école des Mines de Saint-Etienne ont donc été les suivants :

- offrir un modèle sur lequel vont se calquer les définitions du bâtiment et du réseau, ce modèle devant supporter la topologie (adjacence entre pièces), la géométrie (localisation des machines et longueur des câbles), la sémantique (objets graphiques représentant des murs, des câbles, des connecteurs réseau ...),

- permettre différentes visualisations des objets graphiques connus par l'application, et ce, à la demande de l'utilisateur (vue bi-dimensionnelle de chacun des niveaux plans d'un bâtiment, vue détaillée d'une partie d'un niveau, représentation tri-dimensionnelle de l'ensemble des niveaux d'un bâtiment ...),

- présenter un modèle cohérent des interactions graphiques nécessaires à la définition des objets, leur manipulation, la transmission à l'application des mises à jour les concernant,

- mettre à la disposition des utilisateurs des moyens pour dessiner les plans du bâtiment et du réseau et pour effectuer les installations techniques (connecteurs réseau, stations de travail, câbles d'épines dorsales verticales entre étages d'un même bâtiment ...),

- servir de support au mode gestuel, et aussi permettre aux autres modes de l'interface MMI^2 de manipuler des objets graphiques.

Dans ce qui suit, nous présenterons d'abord l'architecture de l'outil de manipulation directe (OMD) et la façon dont s'articulent les différentes entités apparaissant dans cette architecture. Puis, nous établirons une classification des actions de l'utilisateur sur l'OMD et nous présenterons les outils de dessin mis à la disposition des utilisateurs pour leur permettre la saisie de plans. Ensuite, nous établirons le lien entre le modèle basé sur les cartes planaires et la sémantique de l'application. Enfin, nous présenterons les objets sémantiques de l'application et les opérations qui peuvent leur être appliquées.

2.1. Architecture de l'outil de manipulation directe

L'outil de manipulation directe a été développé sur une station Sun du type Sparc_SLC, dotée bien sûr d'un clavier, d'une souris à trois boutons et d'un écran à plusieurs niveaux de gris. Comme cela apparaît dans l'architecture, l'outil de manipulation directe exploite pleinement les possibilités offertes par la librairie du gestionnaire de fenêtres X11 (cf. figure 43).

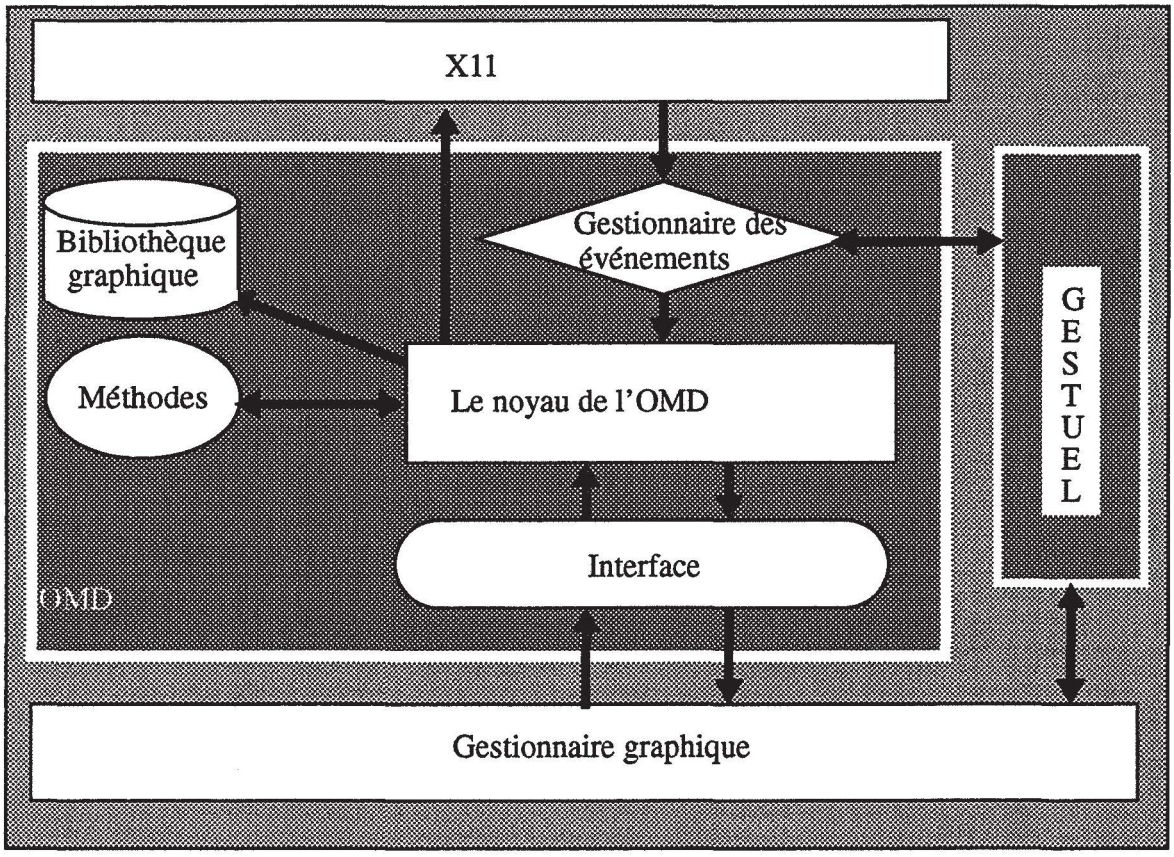


Figure 43 : Architecture de l'outil de manipulation directe

Le gestionnaire de fenêtres X11 a été choisi dès le départ pour son universalité par l'ensemble des partenaires dans le projet MMI^2. En raison de la non disponibilité dès le début du projet du langage de représentation interne CMR, les partenaires ont accepté de représenter de façon interne l'ensemble des interactions par BIM_probe. Ce langage est une couche orientée objet au dessus de Prolog_by_BIM. Comme il n'existait, à l'époque, qu'une version de BIM_probe fonctionnant exclusivement avec le système de fenêtrage Suntools, les premiers développements de l'interface MMI^2, et a fortiori du mode graphique, ont dû être réalisées avec le système de fenêtrage Suntools.

Cette architecture a été voulue très modulaire pour faciliter sa portabilité par rapport au système de fenêtrage. Cette portabilité a été démontrée par le portage effectif de l'outil de manipulation directe du système de fenêtrage Suntools au système de fenêtrage X11. Ces deux systèmes présentent beaucoup de similitudes dans leur fonctionnement et permettent une gestion facile des événements par leur étiquetage : appui sur une touche précise de la souris en un emplacement précis de l'écran, appui sur une touche du clavier alors que le pointeur de la souris est situé à l'intérieur d'une fenêtre précise de l'outil de manipulation directe ...

Le support du gestuel dans le cadre de l'OMD a été géré de façon assez simple : un "bouton basculant" a été prévu dans l'interface de l'OMD. Ce bouton permet de basculer du mode graphique au mode gestuel et inversement. Le traitement des événements, étiquetés au sein de l'OMD par le gestionnaire des fenêtres X11, est alors effectué soit par le mode graphique soit par le mode gestuel, en fonction du mode courant. Nous rappelons qu'en raison de la nature des commandes supportées par le mode gestuel, ce mode n'est utilisé qu'en entrée.

Le mode graphique est, par contre, utilisé en entrée et en sortie. Des interactions combinant par exemple le langage de commande ou une langue naturelle avec le mode graphique constituent d'ailleurs un cas typique de l'accomplissement d'une tâche utilisateur de manière conviviale. Le gestionnaire graphique est là pour servir d'interface avec les autres constituants de l'interface MMI^2, et en premier lieu avec l'interlocuteur privilégié : le contrôleur de dialogue qui joue le rôle de planificateur général et d'interprète vis à vis des autres modes.

En sortie graphique, le noyau de l'outil à manipulation directe se sert d'une bibliothèque graphique permettant d'effectuer des "sorties" élémentaires : tracé texturé d'un segment, remplissage d'un polygone par une texture ... La transition entre les structures de représentation des données spécifiques à l'OMD et celles du gestionnaire graphique est effectuée par des prédicats Prolog. Ces prédicats permettent de véhiculer les informations de l'OMD vers le gestionnaire graphique et aussi d'activer les méthodes (procédures) qui agissent sur les structures de l'OMD.

2.2. Classification des actions de l'utilisateur

Pour utiliser les moyens de dialogue mis à sa disposition dans l'OMD, l'utilisateur se sert des touches de la souris et de sa mobilité, des touches de clavier et des boutons et menus de l'interface graphique de l'OMD.

Dans la suite de ce document, nous allons souvent faire référence à différentes actions de l'utilisateur répértoriées ici et qui peuvent être codées différemment selon le nombre de touches de la souris par exemple.

Dans notre application, nous utilisons une souris à trois touches (gauche, milieu et droite) et nous avons distingué les types d'action suivantes :

adg : désignation d'un emplacement dans l'écran à l'aide de la touche gauche de la souris (appui-relâchement),

add : désignation d'un emplacement dans l'écran à l'aide de la touche droite de la souris (appui-relâchement),

adm : désignation d'un emplacement dans l'écran à l'aide de la touche du milieu de la souris (appui-relâchement).

2.3. Outils de dessin : saisie des plans

Pour permettre une saisie facile des plans, nous avons mis à la disposition des utilisateurs quatre outils de dessin : l'outil de dessin à main levée, l'outil rectangle, l'outil segment et l'outil ligne brisée (cf. figure 44(a)).

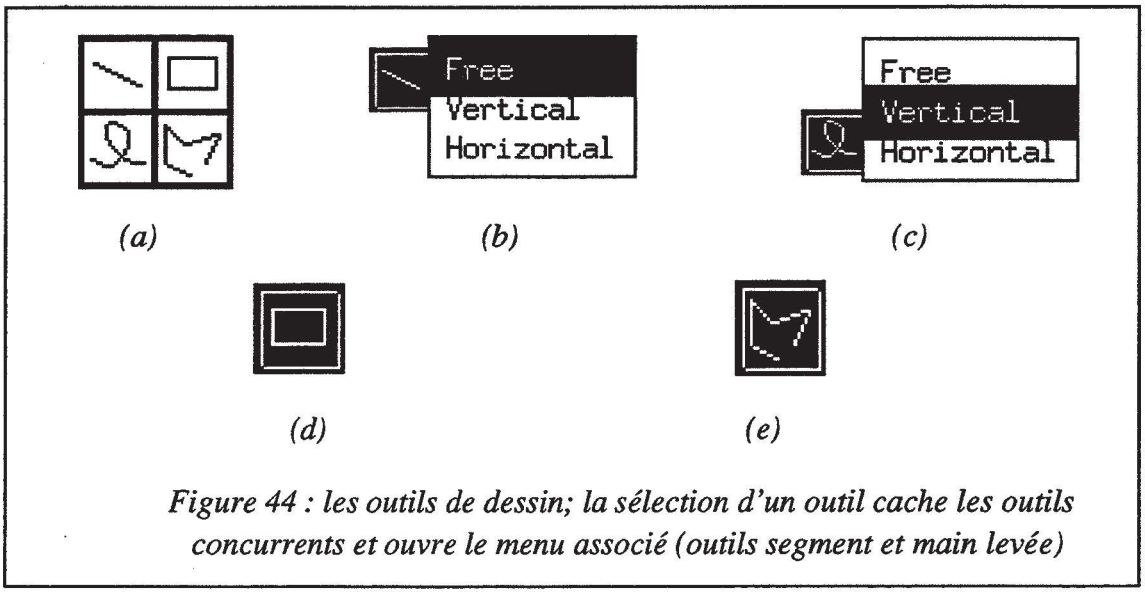


Figure 44 : les outils de dessin; la sélection d'un outil cache les outils concurrents et ouvre le menu associé (outils segment et main levée)

Nous avons prévu une facilité liée au dessin avec l'outil à main levée et l'outil segment : l'utilisateur a la possibilité de contraindre les lignes qu'il trace à l'aide de ces outils à être horizontales ou verticales. Cela s'est traduit dans l'outil de manipulation directe par la mise en place de quatre **boutons** (cf. figure 44(a)), associés chacun à un outil de dessin, et d'un **menu** permettant de contraindre les lignes, dessinées à l'aide des outils à main levée et segment (cf. figures 44(b) et 44(c)) , à être horizontales ou verticales. Ce menu est à trois options : free (direction libre), vertical (direction verticale) et horizontal (direction horizontale).

2.3.1. Fonctionnement des boutons et du menu permettant le dessin

Au lancement de l'outil de manipulation directe, les quatre outils sont visualisés ensemble comme dans la figure 44(a). La **sélection** d'un outil parmi les quatre se fait par une action **adg** à son emplacement. Si l'outil sélectionné a un menu associé, alors ce menu est ouvert (cf. figures 44(b) et 44(c)). Le **choix d'une option de menu** se fait par une action **adg** à son emplacement.

Une action **add** à l'intérieur d'un **menu** provoque la **fermeture du menu**, la **désélection du bouton** courant et l'affichage de tous les **boutons concurrents** du bouton courant : **retour à l'état de la figure 44(a)**. De même, une action **adg** sur un **bouton courant** provoque le **retour à l'état de la figure 44(a)**.

2.3.2. Dessin à l'aide des outils - sémantique du tracé

Chaque ligne dessinée est supposée avoir une sémantique particulière. L'utilisateur peut se la remémorer à l'aide de la texture qu'il a utilisée pour la dessiner. Compte tenu de l'application, deux sémantiques sont proposées à l'utilisateur : la sémantique de bâtiment et la sémantique de réseau. La sémantique de bâtiment est matérialisée par deux textures : une pour les murs perforables et une autre pour les murs non perforables (cf. figures 45(b) et 45(c)). La sémantique de réseau est matérialisée par trois textures différentes de celles de bâtiment : la texture des câbles épais, la texture des câbles fins et enfin la texture de câbles banalisés de réseau (cf. figure 45(d)).

Les sémantiques sont gérées par deux boutons (cf. figure 45(a)), les textures spéciales à chaque sémantique étant gérées par des menus. Ces menus et boutons fonctionnent de manière analogue aux boutons et au menu de dessin.

Le **premier point** d'un tracé est spécifié (placé) dans la zone de dessin par une action adg en son emplacement, et ce, quel que soit l'outil de dessin (**main levée, segment, ligne brisée** ou **rectangle**) utilisé.

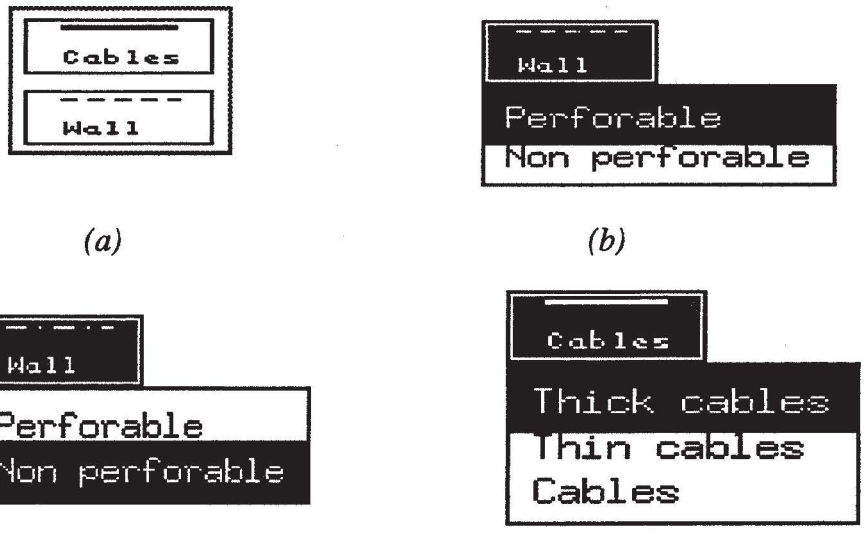


Figure 45 : sémantiques des lignes tracées

Dès l'instant où le placement du **premier point** est **effectué**, l'utilisateur déplace la **souris toutes touches libérées** :

- dans le cas de l'utilisation de l'un des outils **segment** ou **ligne brisée**, l'interface dessine un segment élastique en prenant comme **second point éventuel** le point courant de l'écran,

- dans le cas de l'utilisation de l'outil de dessin à **main levée**, l'interface **emmagasine** pour un traitement ultérieur **tous les points intermédiaires** saisis lors du déplacement de la souris,

- dans le cas de l'utilisation de l'outil **rectangle**, l'interface dessine suite au déplacement de la souris, un rectangle élastique en prenant comme **coin opposé éventuel** le point courant de l'écran.

Lorsque l'outil utilisé est la **ligne brisée**, tout placement de point, à l'aide d'une action adg en son emplacement, après le placement du premier point, crée un **point intermédiaire effectif** de la ligne brisée. L'outil **ligne brisée** est **arrêté** en n'importe quel point intermédiaire désigné par une action add.

Le placement d'un second point par une action adg suffit pour **terminer le dessin** entrepris à l'aide de l'un des outils **segment, rectangle** ou **main levée**.

Une facilité d'utilisation prévue seulement pour les outils **segment** et **rectangle** permet d'**interrompre le dessin** commencé avec ces outils par une action add en n'importe quel point de la zone de dessin (cf. figure 46). Des dessins simples texturés à l'aide de ces outils sont reproduits sur la figure 46.

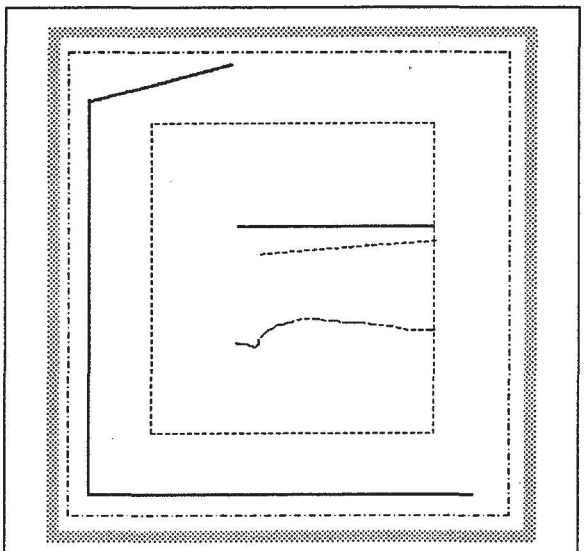


Figure 46 : la zone de dessin

2.3.3. Lien avec le modèle basé sur les cartes planaires

Comme nous l'avons précisé dans le chapitre I, le modèle basé sur les cartes planaires requiert en entrée des segments définis par leur premier et leur second sommet. Nous devons donc traiter les dessins saisis à l'aide des outils de dessin avant de les communiquer aux algorithmes des cartes planaires. Dès la saisie, nous stockons les dessins, faits à l'aide des outils précédemment décrits, dans une liste doublement chaînée de structures *Point :*

```
struct Point
    {
        struct Point *avant;
        struct Point *après;
        int x;
        int y;
        int outil;
        int texture_ligne;
        int largeur_ligne;
        boolean fin_outil;
    }
```

L'étiquetage de chaque point par ses coordonnées écran est fait par le gestionnaire de fenêtres. Deux variables globales de type point recueillent les listes des points associés respectivement au bâtiment et au réseau. La distinction entre bâtiment et réseau est facilitée d'une part par le champ *fin_outil* qui indique, pour chaque point, la fin ou non de l'utilisation courante de l'outil *outil* en ce point. D'autre part, la connaissance du bouton de sémantique activé (cf.

figures 41(a) et 56(a)) permet d'attribuer chaque occurrence d'utilisation d'outil (champ fin_outil) à une des deux listes de points. Les champs *texture_ligne* et *largeur_ligne* sont stockés en chaque point pour permettre l'affichage immédiat, après chaque saisie, de tous les bipoints d'une même occurrence d'outil en respectant la texture et l'épaisseur de la sémantique associée à l'utilisation de l'outil de dessin *outil*.

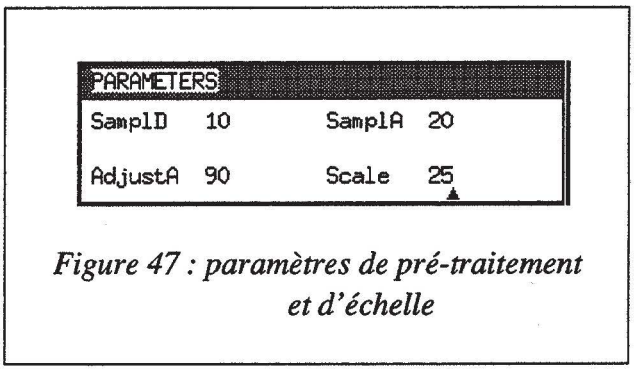


Figure 47 : paramètres de pré-traitement et d'échelle

Le traitement du tracé initial avant sa structuration à l'aide de la notion de carte planaire est regroupé sous le nom de pré-traitement. Ce pré-traitement est effectué en trois étapes : **échantillonnage**, **recalage** et **élimination des ratures**. L'outil de manipulation directe affiche, dans une zone de la fenêtre principale d'interaction graphique, les paramètres courants de pré-traitement ainsi qu'un paramètre d'échelle (cf. figure 47).

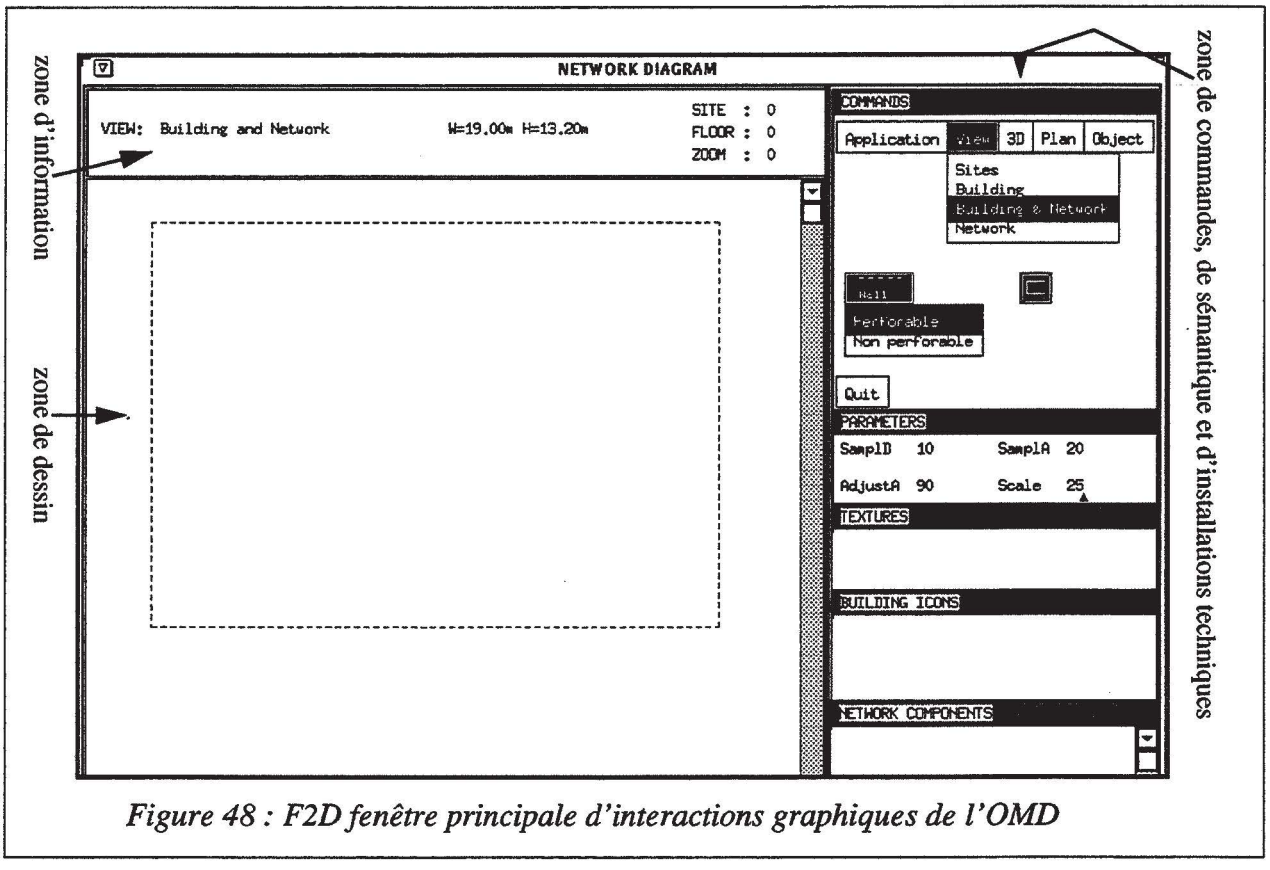


Figure 48 : F2D fenêtre principale d'interactions graphiques de l'OMD

Les paramètres SamplD et SamplA sont utilisés pour l'échantillonnage, le paramètre AdjustA est utilisé pour le recalage. Le paramètre Scale concerne l'échelle et permet à l'utilisateur d'adapter celle-ci en fonction de la géométrie du tracé et de la surface de la zone de dessin (cf. figure 48).

Dans la première description d'un plan, c'est à dire au niveau de détail 0, la longueur réelle en mètres des segments est affichée dans la zone d'information (cf. figure 48). La longueur réelle en mètres d'un segment correspond à sa longueur en pixels divisée par le paramètre Scale.

a. L'échantillonnage

L'échantillonnage consiste en la réduction du nombre de points saisis. En effet, lors du dessin à main levée, la souris transmet un nombre important de points que nous stockons dans des structures *Point*.

Plusieurs techniques d'échantillonnage de tracés bruts ont été développées par Berthod et Jancenne [BERT 79]. Nous avons choisi la méthode exposée dans [BELA 82]. Cette méthode permet de représenter une liste de points, comme indiqué dans la figure 49, par les deux points extrêmes donnant ainsi lieu à la création d'un segment.

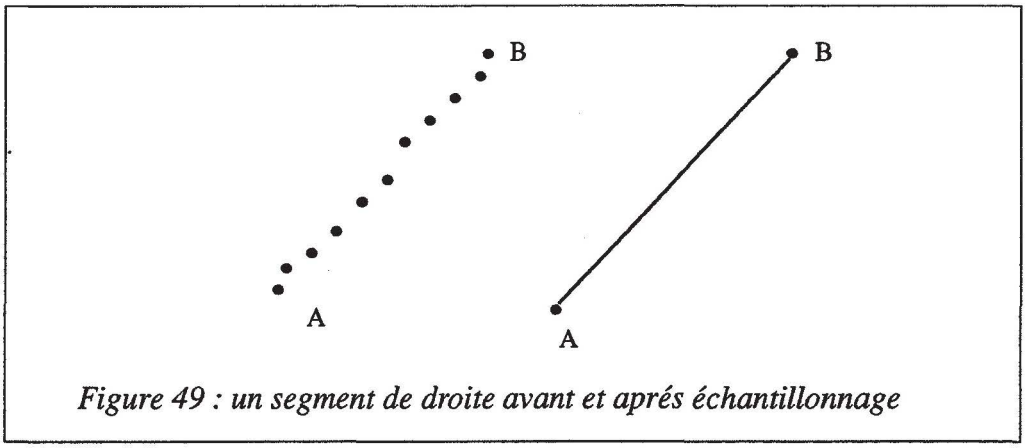


Figure 49 : un segment de droite avant et aprés échantillonnage

Or, pour une ligne de direction donnée, les points qui la composent ne sont que rarement alignés. Des déformations (pics) peuvent être involontairement introduits par l'utilisateur suite au tremblement de la souris dans sa main. La correction des pics doit se faire en conservant les changements de direction volontaires de l'utilisateur (cf. figure 50). L'algorithme permettant d'effectuer l'échantillonnage utilise les paramètres fournis par l'utilisateur : SamplA angle minimum toléré entre deux traits consécutifs; et SamplD longueur (en pixels) minimum tolérée pour un trait. L'étude faite par Belaid et Masini [BELA 82] montre qu'une valeur de SamplA comprise entre 15 et 20 degrés donne de bons résultats.

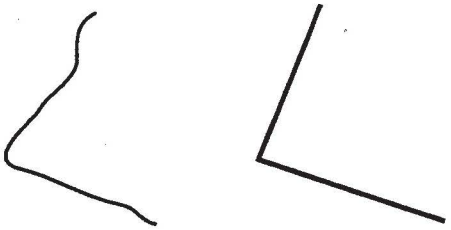


Figure 50 : Tracé déformé avant et après échantillonnage

b. Le recalage

L'étape de recalage prend en entrée la liste des points échantillonnés. Le but de cette étape est principalement d'assurer l'orthogonalité et le parallélisme des segments de droite, et plus généralement de respecter les désirs de l'utilisateur en termes de directions angulaires à suivre. La première direction angulaire est celle correspondant à l'horizontale orientée vers la droite (angle 0 degrés). Ensuite, cet angle est incrémenté à chaque fois de la valeur du paramètre AdjustA pour essayer de recaler un segment donné sur l'une des directions angulaires qui l'encadrent. Le recalage se fait par rapport à la direction angulaire la plus proche en conservant le milieu du segment initial (cf. figure 51). L'algorithme que nous avons utilisé pour le recalage a été proposé par Michelucci [MICH 84a].

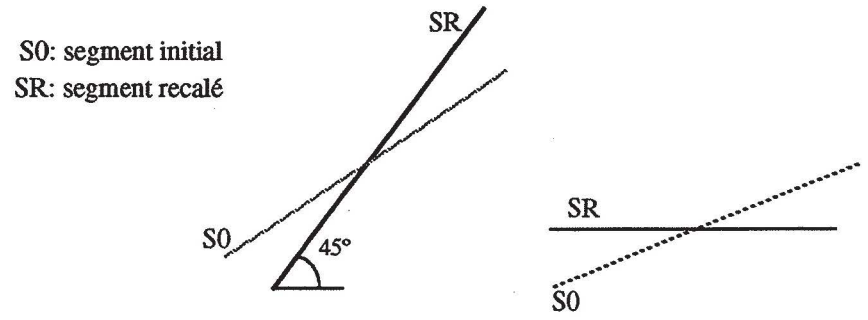


Figure 51 : recalage à 45 degrés

c. L'élimination des ratures

Les ratures sont du type rupture d'un tracé, superposition de deux tracés ... (cf. figure 52). Or, ce genre de rature ne peut être corrigé sans connaissances contextuelles comme souligné dans [HERO 76].

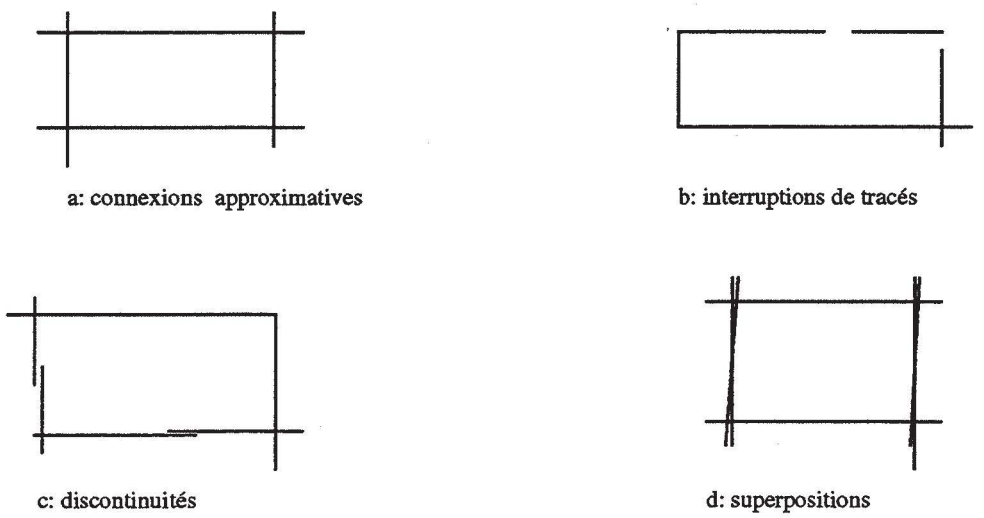


Figure 52 : exemples de ratures

Nous éliminons ce type de rature lors de la construction de la carte planaire, à l'aide de méthodes heuristiques. Concernant le cas de la figure 52(a), cela revient par exemple à éliminer toutes les arêtes pendantes (cf. définiton dans le chapitre I). Les cas de rature des figures 52(c) et 52(d) sont éliminés automatiquement par l'algorithme de construction des cartes planaires qui permet l'élimination des arêtes superposées et la fusion des arêtes de discontinuité. Certaines parmi les ratures du type de la figure 52(b) sont éliminées par la considération d'un cercle de rayon fixé à l'avance : les extrémités des segments se trouvant à l'intérieur de ce cercle sont raccordées en un point situé dans le cercle (cf. figure 53). Cette méthode peut être appliquée également au cas de la figure 52(a).

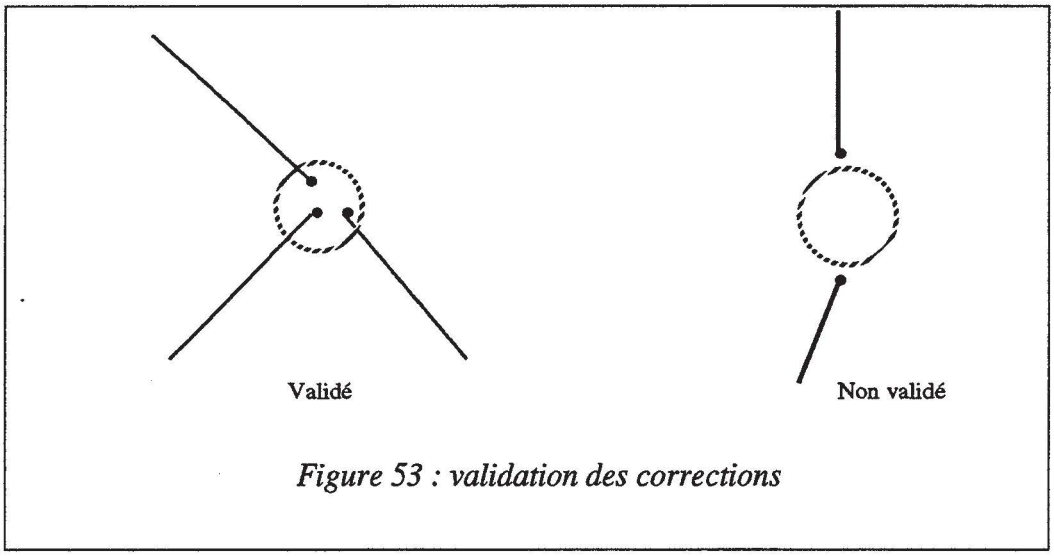


Figure 53 : validation des corrections

Dans la pratique, l'identification des sommets à raccorder se fait dans la procédure de création des sommets. En effet, la création d'un nouveau sommet entraine le parcours de ceux qui existent déjà en vue d'une insertion correcte dans le x-y-ordre. Nous profitons de ce parcours pour ne faire de création effective de sommet que si le sommet à créer est distant, de tous ceux qui existent, d'au moins cinq pixels. Cinq pixels correspondent à une distance entre deux points écran raisonnablement discernables l'un de l'autre par un utilisateur. Le raccordement des sommets 'voisins' est effectué dans l'étape d'initialisation et dans les algorithmes de construction statique et incrémentale des cartes planaires.

d. Exemple d'une carte planaire construite après pré-traitement d'un dessin initial

La figure 54(a) montre un dessin initial avec toutes ses malformations. Après un pré-traitement, l'algorithme de construction statique de carte planaire a été lancé (cf. chapitre I). La carte planaire globale est faite d'un ensemble de faces. Les faces internes (faces) ont la sémantique *Pièce* d'un étage par défaut. Chaque face est donc associée à un objet sémantique caractérisé par un identificateur entier. Sur la figure 54(b) ces identificateurs sont marqués dans le coin bas gauche de chaque face.

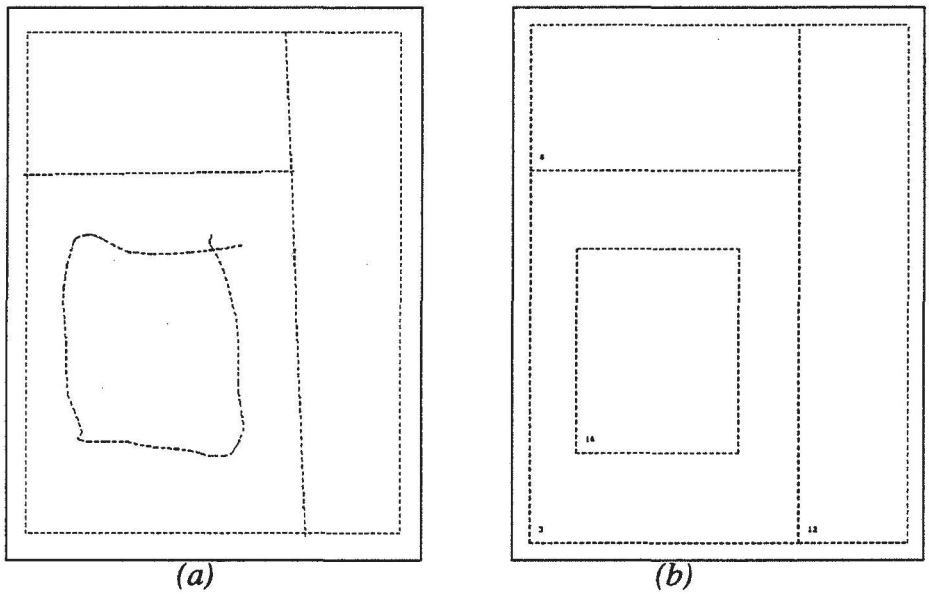


Figure 54 : un dessin initial avant et après les algorithmes de pré-traitement et de construction statique de carte planaire

2.4. Cartes planaires et sémantique de l'application

2.4.1. Sémantique du modèle $2D^{1/2}$ basé sur les cartes planaires

Nous rappelons ici que l'application envisagée au départ pour valider l'ensemble des modes du projet MMI^2 est une base de connaissances dans le domaine de la conception et de l'analyse de réseaux informatiques (NEST).

a. La structure de données $2D^{1/2}$

Un réseau informatique est constitué essentiellement de câbles, de serveurs, de stations de travail et d'éléments techniques tels que des gaines, des connecteurs ... L'ensemble des constituants peut être installé sur plusieurs étages d'un même bâtiment, ou sur plusieurs bâtiments voisins englobant ainsi la notion de site. Nous rappelons que l'expertise recueillie dans le cadre de NEST est spécifique aux technologies Ethernet et TCP/IP.

Les plans saisis dans le cadre de cette application sont donc représentés structurellement par le modèle $2D^{1/2}$ présenté dans le deuxième chapitre de cette thèse. Un bâtiment est constitué de plusieurs étages. Il est englobé dans la structure *Liste_carte* liste doublement chaînée de représentations d'étages. Chaque étage est représenté par un tableau à deux éléments de type carte planaire : CP1 représentant la sémantique de bâtiment et CP2 celle de réseau. La structure *Liste_carte* adaptée à l'application est obtenue en remplaçant **k** par **2** dans celle présentée dans le chapitre II de ce document.

La carte planaire CP1 de bâtiment (tcarte[0]) regroupe donc des objets du type mur, pièce, escalier ... ; la carte planaire CP2 de réseau (tcarte[1]) regroupe des objets du type segment de câble, station de travail ... Un objet de type mur peut avoir la sémantique de mur perforable ou de mur non perforable. Ce choix est fait une première fois lors de la création. Une fois un mur créé, il peut être reclassifié, c'est à dire voir sa sémantique passer de mur perforable à mur non perforable, ou l'inverse. Les textures utilisées pour la visualisation des murs perforables et des murs non perforables ne sont pas les mêmes. Ceci pour faciliter la lecture des informations à l'écran. La distinction des textures n'empêche pas que les arêtes représentant des murs perforables et des murs non perforables d'un même niveau plan soient structurées au sein d'une même carte planaire, associée à la sémantique de bâtiment.

Chacune des cartes planaires CP1 et CP2 possède un champ de type *Objet* qui pointe sur la structure *Objet* correspondant au dernier objet créé et qui lui a été attribué. La structure de données a été présentée section B.3.2. du chapitre II de ce document.

b. La sémantique associée aux cartes planaires

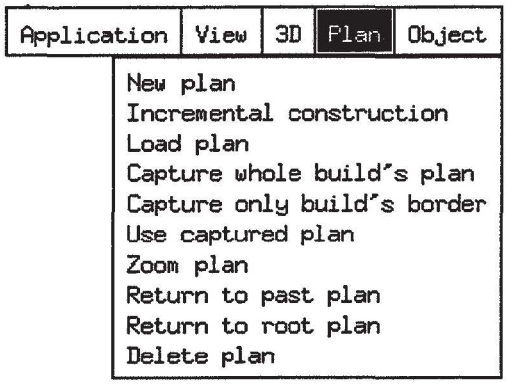
Nous avons opté pour la séparation entre la sémantique et la topologie. En effet, l'utilisateur doit pouvoir manipuler librement les entités qui ont un sens pour lui sans se soucier du modèle de représentation sous-jacent. La structure *cplanaire* contient donc d'une part la topologie du plan sous la forme : arborescence des faces, incidence entre arêtes et ordre lexicographique des sommets. D'autre part, cette structure contient une liste chaînée d'objets sémantiques associés chacun à une entité physique pour l'utilisateur. Les entités issues de la construction de la carte planaire ayant été jugées nécessaires dans le cadre de l'application sont les arêtes (pour CP1 et CP2), les faces internes (essentiellement pour CP1 : il n'y a pas de boucle dans les réseaux Ethernet), et les faces externes (pour CP2; pour CP1 connexe, la face la plus externe est associée à un objet intitulé *niveau*). En fait, cet objet niveau et les autres (murs, pièces, couloirs ...) devaient être en harmonie avec la décomposition du monde de l'application en hiérarchies d'objets. Outre ces objets issus de la construction des cartes planaires, d'autres objets techniques représentés par des textures (faux-plafond, faux-plancher, gaine technique) ou des icônes (serveurs, connecteurs réseau, sorties d'escalier ou d'ascenseur ...) s'ajoutent à ceux de CP1 ou CP2 selon qu'ils sont de sémantique bâtiment ou réseau. L'installation de certains connecteurs de réseau intervient de façon incrémentale dans la carte planaire de réseau en provoquant l'éclatement de segments de câble (icônes de connexion de réseau RCON du chapitre 2).

Les câbles en tant qu'objets sont délimités par des connecteurs. Deux arêtes de réseau non colinéaires représentent graphiquement le même objet câble du moment qu'aucun connecteur n'est installé au sommet de jointure entre les deux arêtes. De ce fait, chaque arête de la carte planaire de réseau est associée à un objet dont la sémantique est partie_de_câble. Toutes les parties de câble situées entre deux connecteurs RCON constituent un seul objet dont la sémantique est câble. Chaque objet de type câble est lié à ses constituants par des connexions typées 'partie_de'. Ces connexions (structure Liens) ont été présentées section B.2.1. du chapitre II de ce document.

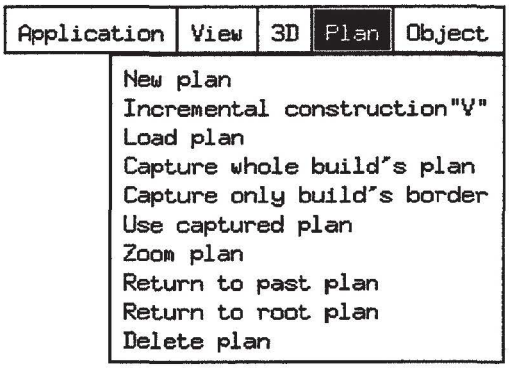
Cette même structure de connexion permet de maintenir le sens global des données réparties dans deux cartes planaires indépendantes. Par exemple, elle établit des relations du type : le serveur est dans la pièce 11, la gaine technique contenant le câble longe les murs m17, m18 et m21 ...

c. Les aspects liés à l'outil de manipulation directe

Un utilisateur peut décrire, l'un après l'autre, tous lès étages constituant le bâtiment où est installé le réseau. Il se sert essentiellement des outils de dessin et de sémantique décrits dans la section 2.3. de cette partie. Après avoir dessiné à l'aide des outils une esquisse du bâtiment et du réseau (cf. figure 46), l'utilisateur peut lancer la construction statique d'une carte planaire qui créera du même coup la sémantique des deux cartes planaires CP1 et CP2. Pour cela, l'utilisateur doit activer le bouton de l'outil de manipulation directe intitulé 'Plan' et choisir l'option de menu 'Incremental construction' pour basculer après en construction incrémentale de ce niveau plan (cf. figures 55(a) et 55(b)). Suite à cette opération, tout ajout de dessin, s'effectuera incrémentalement après pré-traitement. Le bouton 'Plan' fait partie d'un ensemble de boutons intitulé 'barrette'.



(a) état de construction statique de plans

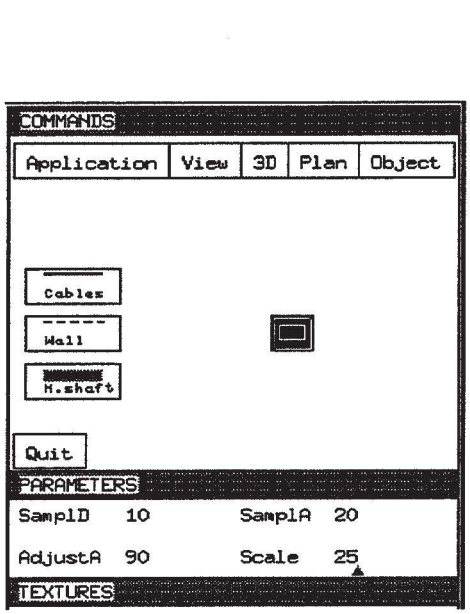


(b) état de construction incrémentale de plans

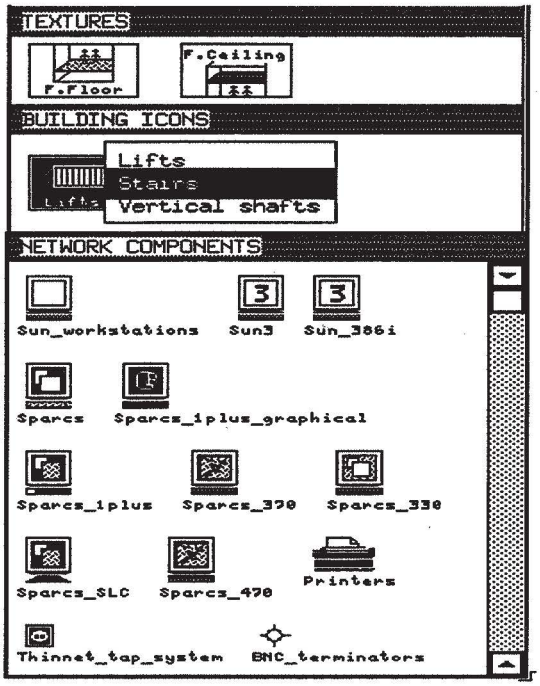
Figure 55 : bouton 'Plan' activé parmi ceux de la 'barrette'

Ce n'est que dans l'état de construction incrémentale que les objets techniques sont rendus visibles à l'utilisateur et accessibles en utilisation : icônes de réseau, icônes de cage d'ascenseur ou d'escalier (service inter-étage), texture matérialisant l'installation de faux-plancher ou de faux-plafond, texture de gaine technique horizontale (cf. figure 56).

Les objets techniques associés à la carte planaire de réseau d'un niveau se limitent à ceux représentés graphiquement par des icônes de réseau. Tous les autres sont intégrés à la carte planaire de bâtiment. Nous saisissons l'occasion pour préciser qu'une partie de câble d'épine dorsale verticale, passant par deux sorties de service inter-étage d'étages consécutifs, n'est attribuée à aucun des deux étages reliés : elle est représentée par une connexion du type de la figure 14, marquée par un type spécial et liant les deux objets de service inter-étage à travers lesquels passe le câble.



(a) 'H.shaft' sémantique de lignes brisées. Elle n'est utilisable qu'en incrémental, dans la vue bâtiment et réseau, pour créer les objets : gaine technique horizontale.

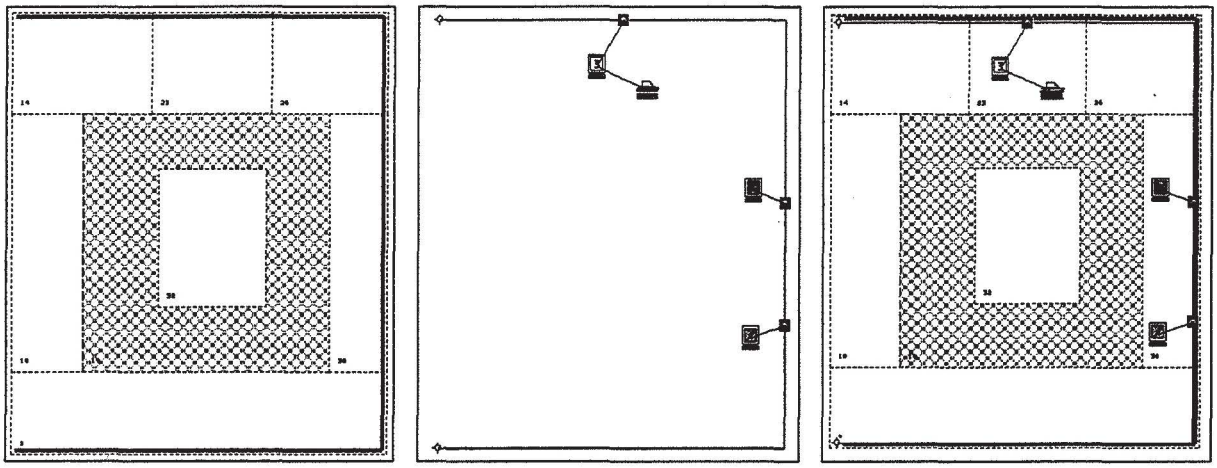


(b) autres installations techniques qui ne sont accessibles que dans l'état incrémental

Figure 56 : fonctionnalités d'installations techniques qui ne sont accessibles et visibles qu'en état de construction incrémentale de carte planaire

Après avoir décrit un étage, l'utilisateur peut décider d'en créer un autre en activant l'option de menu intitulée 'New plan', menu attaché au bouton 'Plan' de la barrette (cf. figure 55). L'utilisateur est alors guidé dans l'attribution d'un numéro à ce nouvel étage par rapport à ceux existants. Ce guidage est fait à l'aide d'une fenêtre de message. Une facilité supplémentaire est proposée à l'utilisateur lui permettant de dupliquer l'ensemble du dessin de bâtiment d'un étage, ou uniquement son pourtour externe, pour le reprendre dans un autre étage. Cela fonctionne de la manière suivante : l'utilisateur charge à l'écran un étage qu'il a précédemment décrit à l'aide de l'option de menu 'Load plan' et d'un guidage par fenêtre de message; ensuite, il choisit l'une des options 'Capture whole build"s plan' ou 'Capture only build"s border', les tracés désirés se trouvent ainsi capturés; enfin, l'utilisateur active la fonctionnalité permettant de créer un nouvel étage, suite à quoi il choisit l'option 'Use captured plan' du menu du bouton 'Plan'. Les autres options du menu associé au bouton 'Plan' seront abordées ultérieurement.

Du point de vue de l'affichage, une zone est réservée au dessin dans la fenêtre principale des interactions graphiques (cf. figures 46, 48 et 57). L'utilisateur peut choisir de visualiser dans cette zone, selon son désir, soit le plan de bâtiment seul (figure 57(a)), soit le plan du réseau seul (figure 57(b)), soit les deux plans de bâtiment et de réseau superposés (figure 57(c)). Nous tenons à préciser que la texture des figures 57(a) et 57(c) représente un faux-plafond installé dans une des pièces de l'étage visualisé et que les objets faux-plafond et faux-plancher sont attachés à la sémantique de bâtiment.



(a) plan de bâtiment *(b) plan de réseau* *(c) plans de bâtiment et de réseau superposés*

Figure 57 : différentes vues d'un étage

Les différentes vues bi-dimensionnelles des plans d'un étage sont gérées par un menu associé à un bouton de la barrette intitulé 'View' (cf. figure 58).

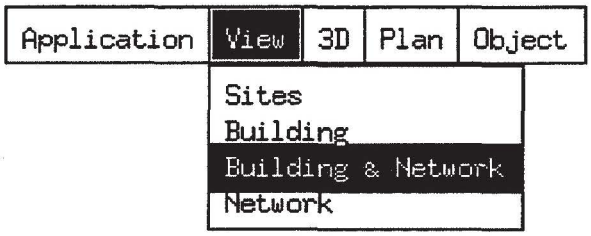
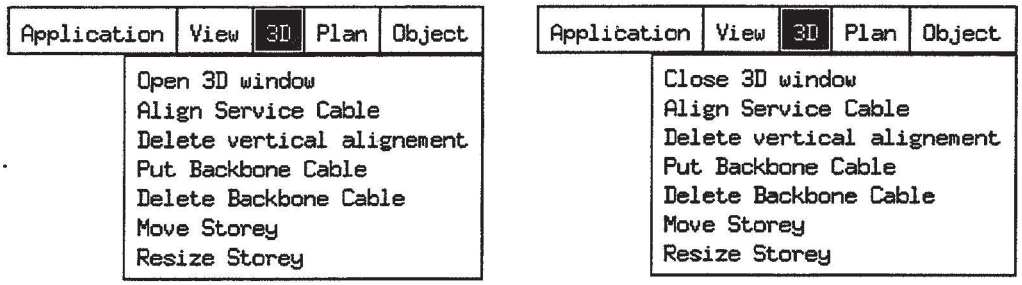


Figure 58 : le bouton 'View' et le menu associé gérant les différentes vues

Quant à la vue tri-dimensionnelle de l'ensemble des étages constituant un bâtiment, ainsi que l'ensemble des fonctionnalités inter-étages, elles sont gérées par un menu associé au bouton de la barrette intitulé '3 D' (cf. figure 59).



(a) F3D fermée (b) F3D ouverte

Figure 59 : le bouton '3D' et le menu regroupant les fonctionnalités inter-étages

L'ouverture et la fermeture de la fenêtre F3D, dédiée à la visualisation tri-dimensionnelle des étages, sont commandées par la première option du menu associé au bouton de la barrette intitulé '3D'. Cette option de menu bascule de la commande 'Open 3D window' à la commande 'Close 3D window', et ce en fonction de l'état fermé ou ouvert de F3D. Les autres fonctionnalités gérées par le menu '3D' seront décrites ultérieurement. Ci-dessous, un exemple d'affichage tri-dimensionnel d'un immeuble à plusieurs étages (cf. figure 60). La représentation tri-dimensionnelle permet de visualiser uniquement trois étages consécutifs contenant l'étage en cours de finalisation dans la fenêtre principale d'interaction graphique F2D. La visualisation des étages non affichés du même immeuble est commandée par l'ascenseur associé à F3D.

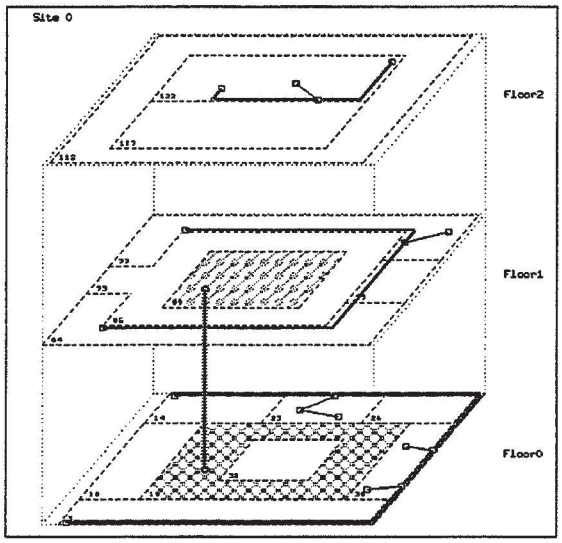


Figure 60 : représentation tri-dimensionnelle d'un immeuble

La fenêtre F3D a été partagée en trois zones, chacune d'elles devant contenir le tracé tri-dimensionnel d'un niveau plan. Il en découle que l'espace reservé à l'affichage tri-dimensionnel est nettement réduit par rapport à celui de l'affichage bi-dimensionnel. La zone reservée à l'affichage tri-dimensionnel d'un niveau risque d'être surchargée d'informations et donc illisible. Nous avons opté pour une réduction de la taille de toutes les données icôniques. Nous avons associé une forme icônique réduite à chacune des sémantiques prises en compte dans l'application. Toutes les icônes voient leur représentation uniformisée, en quelque sorte, dans la fenêtre tri-dimensionnelle. Le centre de chaque représentation d'icône dans F3D est exactement le transformé tri-dimensionnel du centre de l'icône bi-dimensionnelle représentée. Cette uniformisation permet d'avoir une meilleure vue générale, sachant que les détails concernant les icônes peuvent être obtenus à l'aide d'une fonction de sélection et d'une fonction permettant d'avoir des informations sur des objets préalablement sélectionnés.

Les variations relatives de dimension et de positionnement entre les dessins des différents niveaux plans sont mis en valeur par une représentation en pointillé de l'image tri-dimensionnelle de la boîte rectangulaire englobant tous les dessins des niveaux plans. Des verticales, joignant les coins des différentes visualisations de cette boîte dans la fenêtre tri-dimensionnelle, sont matérialisées en pointillé pour donner une meilleure impression tri-dimensionnelle.

De même que dans la fenêtre F2D, l'utilisateur peut choisir de visualiser sur F3D les plans de bâtiment des étages seuls, les plans de réseau des étages seuls ou les plans de bâtiment et de réseau des étages superposés (cas de la figure 60). Ainsi, nous avons deux types de superposition : deux objets d'un même étage pourront avoir une relation du type : la machine M1 est dans la pièce P4 tandis que des objets appartenant à des niveaux plans strictement parallèles pourront avoir des relations du type : la machine M3 est au dessus de la pièce P5.

2.4.2. Sémantique de la hiérarchie dans le modèle basé sur les cartes planaires

Une des raisons essentielles de l'introduction de la hiérarchie dans le modèle basé sur les cartes planaires est de permettre à l'utilisateur d'avoir différentes vues de ses plans, vues associées à des points de vue différents. Nous avons distingué deux types de hiérarchies : une hiérarchie qui permet de modéliser un site de plusieurs bâtiments et d'avoir une vue globale sur le site; et une autre hiérarchie permettant à l'utilisateur de mieux gérer l'espace écran et de décrire ses plans au niveau de détail désiré sans surcharger pour autant l'écran de l'ordinateur.

a. La structure de données hiérarchique

La modélisation d'un site constitué de plusieurs bâtiments s'est faite de façon assez naturelle. D'une part, la visualisation d'un ensemble de bâtiments se fait assez classiquement sur un plan 'site'. D'autre part, le tracé d'un immeuble décrit en partie par ses étages peut être assimilé, d'un point de vue global, à celui du pourtour de son plus bas étage (face externe de la carte planaire

de bâtiment du niveau 0), reproduit à une certaine échelle, centré en un point donné et avec son axe horizontal tourné d'un certain angle par rapport à l'axe horizontal du plan 'site'.

La structure de données qui supporte la notion de site dans notre application est la suivante :

```
struct L_IMMEUBLE
        {
                struct Liste_carte *rez_de_ch;
                int no_imm;
                int x_centre;
                int y_centre;
                int angle_imm;
                struct L_IMMEUBLE *imm_avant
                struct L_IMMEUBLE *imm_après;
        }
```

L'accès à toutes les données d'un immeuble se fait via la structure *Liste_carte* représentant son rez de chaussée, assimilé au niveau le plus bas (de numéro 0).

La structure *Liste_carte* qui permet de prendre en compte l'aspect hiérarchique dans la représentation d'un même plan bi-dimensionnel est celle présentée dans la section 2.5. de la partie C du chapitre II, avec un nombre de sémantiques k=2.

b. La sémantique associée aux hiérarchies de cartes planaires et au point de vue 'site'

La sémantique associée au point de vue 'site' est assez rudimentaire. Cela correspond à peu près à une vue aérienne des immeubles qui le constituent; en effet, dans cette vue les pourtours des étages supérieurs d'un immeuble ne sont pas pris en compte dans la vue 'site'.

Par contre, la sémantique associée à la hiérarchie des cartes planaires est beaucoup plus développée. Au départ, l'utilisateur décrit, à un certain niveau de détail, le bâtiment et le réseau d'un étage.

Par rapport à ce qui a été décrit dans la partie C du chapitre II, le bâtiment correspond à la sémantique de base et le réseau informatique correspond à une sémantique de réseau particulière. Les objets antérieurement typés ab correspondent à des murs (mk) et ceux antérieurement typés pab correspondent à des partie_de_mur (pmk). Les objets antérieurement typés ar correspondent à des partie_de_cable (pck) et ceux antérieurement typés l_ar correspondent à des câbles.

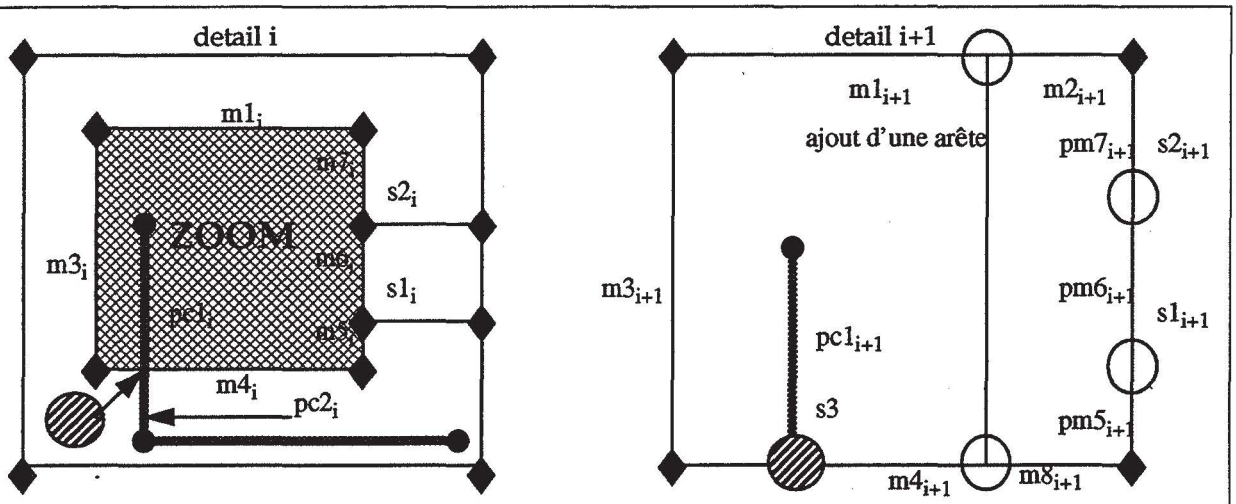


Figure 61 : sémantique associée aux hiérarchies de 'détail'

Supposons que l'utilisateur choisisse de détailler la pièce texturée dans la figure 61. Le câble de réseau traversant cette pièce est coupé symboliquement à sa traversée. Cela ne nuit aucunement à la sémantique de départ qui utilise déjà la notion de partie de câble : les segments texturés, horizontal et vertical dans le détail i, sont, chacun, associés à une partie de câble et les deux objets qu'ils représentent sont liés par des connexions du type de ceux de la figure 14 à un objet de type câble.

Toutes les connexions dont nous parlerons dans la suite de ce document sont assurées par la structure *Liens* représentée figure 14. Pour assurer une meilleure correspondance géométrique et sémantique lors du lancement de l'opération 'Zoom' et lors d'ajouts ultérieurs dans le niveau i+1, nous avons mis en place des éléments de continuité (cf. partie C du chapitre 2).

Pour le cas de la figure 61, nous introduisons un élément de continuité de réseau dans le niveau i et un autre dans le niveau i+1. Ces éléments sont liés entre eux par une connexion marquée 'détail' et sont situés aux emplacements de la traversée de la face détaillée par le câble au niveau i et au niveau i+1. L'introduction d'un élément de continuité de réseau, ayant des propriétés analogues à celles des connecteurs de réseau, entraîne donc l'éclatement de $pc1_i$ en $pc1_i$ et $pc2_i$. $pc2_i$ n'a aucun lien 'détail' avec le détail i+1 représenté sur la figure 61 mis à part un passage détourné par l'élément de continuité de réseau. Par contre, $pc1_i$ est lié par une connexion marquée 'détail' à $pc1_{i+1}$ homologue exact de $pc1_i$ dans le niveau de détail i+1. Le câble défini par $pc1_{i+1}$ peut alors être connecté par extension à celui contenant $pc1_i$.

Il en est de même pour le bâtiment où, dès le lancement de l'opération 'Zoom', des éléments de continuité de bâtiment sont créés et placés en $s1_i$ et $s2_i$ et $s1_{i+1}$ et $s2_{i+1}$. Ils sont représentés sur la figure 61 par des cercles non texturés. Ceux placés en $s1_{i+1}$ et $s2_{i+1}$ donnent naissance à des partie_de_mur $pm5_{i+1}$, $pm6_{i+1}$, $pm7_{i+1}$ constituant un mur m10 (via des

connexions marquées partie_de). Ces mêmes objets partie_de_mur sont liés par une connexion marquée 'détail' à des murs du niveau i. En somme, dès le lancement de l'opération 'Zoom', nous avons les liens de détail : ($m1_i$, $m1_{i+1}$), ($m3_i$, $m3_{i+1}$), ($m4_i$, $m4_{i+1}$), ($m5_i$, $pm5_{i+1}$), ($m6_i$, $pm6_{i+1}$), ($m7_i$, $pm7_{i+1}$). L'ajout d'une arête verticale coupant $m1_{i+1}$ et $m4_{i+1}$ provoque la création de nouveaux objets, dont deux éléments de continuité de bâtiment qui pourront être exportés au niveau i à la demande de l'utilisateur. Une mise à jour des connexions 'détail' est également effectuée : $m1_i$ se retrouve lié à deux murs $m1_{i+1}$ et $m2_{i+1}$ qui le détaillent.

c. Les aspects liés à l'outil de manipulation directe

Au lancement de l'outil de manipulation directe, une structure de données 'site' est instanciée. Elle contient un immeuble de numéro 0. L'utilisateur voit affichés les outils et la zone de dessin lui permettant de décrire de façon bi-dimensionnelle le niveau 0 de l'immeuble 0. L'utilisateur peut choisir de définir complètement ou partiellement l'immeuble 0 avant de passer à la vue 'site' pour créer d'autres immeubles, ou choisir de passer d'abord à la vue 'site' pour créer d'autres immeubles et les sélectionner dans l'ordre désiré en vue d'une description de bâtiment comme nous l'avons décrit dans la section 2.4.1.c de cette partie.

L'utilisateur passe à la vue 'site' en activant la première option du menu associé au bouton 'View' de la barrette (cf. figure 58). La création de nouveaux sites (immeubles) s'effectue comme l'installation des icônes de réseaux (cf. paragraphes ultérieurs) après choix de la première option du menu associé au bouton 'Site' de la barrette. Nous précisons que dès le choix par l'utilisateur de la vue 'site', les boutons composant la barrette sont mis à jour pour ne faire apparaître que ceux utiles pour cette vue (cf. figure 62).

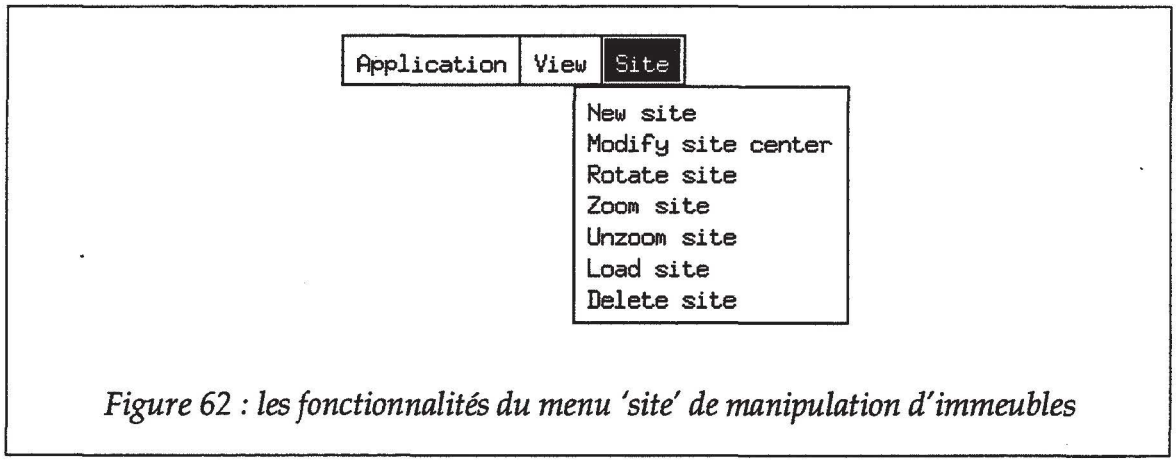


Figure 62 : les fonctionnalités du menu 'site' de manipulation d'immeubles

Les fonctionnalités relatives aux immeubles et groupées dans le menu associé au bouton 'Site' sont assez simples. La modification des coordonnées du centre d'un immeuble (celui de son étage de numéro 0 : RdC) et la rotation de l'axe horizontal du dessin de son rez de chaussée par rapport à celui du plan 'site' sont guidées par la fenêtre de message et la zone des paramètres. Le contenu de la zone des paramètres de l'outil de manipulation directe change suivant le contexte (cf. figures 47 et 63).

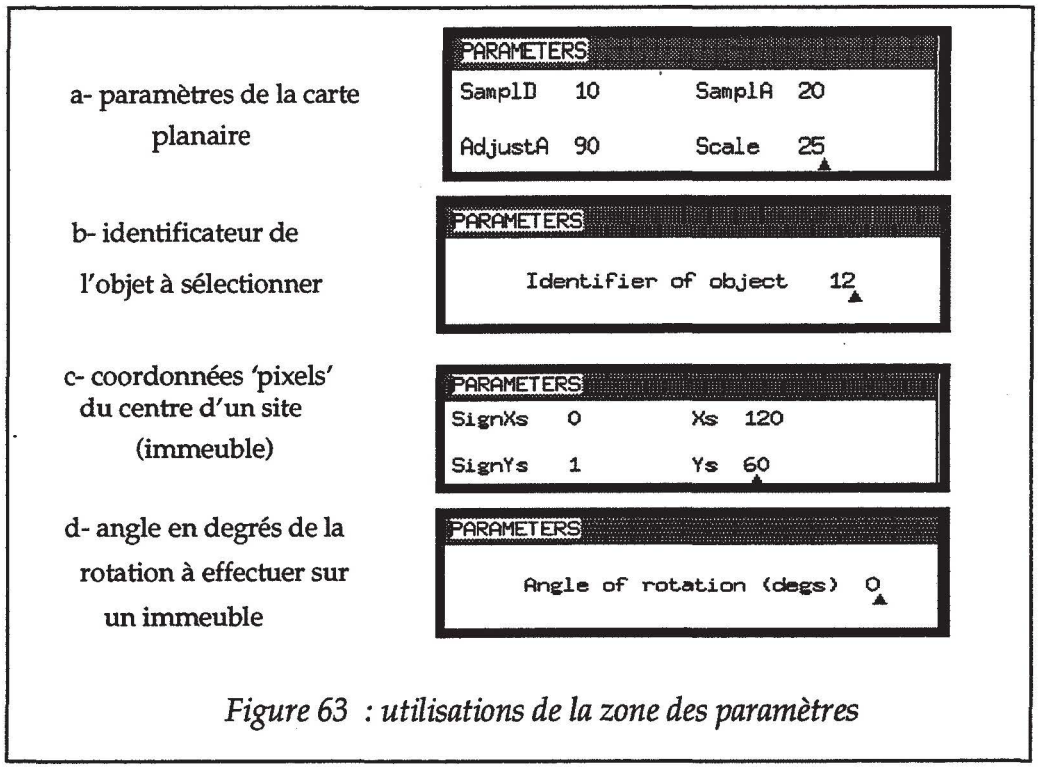


Figure 63 : utilisations de la zone des paramètres

Les options du menu 'Site' intitulées 'Zoom site' et 'Unzoom site' permettent de reproduire le pourtour du rez de chaussée d'un immeuble au 1/2, 1/4, 1/8 ou 1/16. L'ordre de zoom 0 correspond à une représentation sous forme icônique. Cet ordre est incrémenté ou décrémenté selon l'option choisie après la sélection d'un immeuble.

En ce qui concerne la hiérarchie des cartes planaires, elle est gérée par des options du menu associé au bouton 'Plan' de la barrette (cf. figure 55).

Les options de menu 'Return to past plan' (resp. 'Return to root plan') permettent d'aller d'un niveau de détail i+1 au niveau de détail i (resp. au niveau de détail 0). Par contre, au niveau de détail i, l'option de menu 'Zoom plan' fonctionne de deux façons :

- soit l'utilisateur sélectionne des faces du niveau i et active cette option; si un zoom associé existe, l'outil de manipulation directe (OMD) affiche les plans i+1 associés; si aucun zoom associé n'existe, l'OMD regarde si l'une des faces sélectionnées figure parmi celles d'un autre détail i+1 : dans l'affirmative, il annule l'opération de 'Zoom' et alerte l'utilisateur par message; dans la négative, l'OMD crée un niveau de détail i+1 associé aux faces sélectionnées,

- soit l'utilisateur ne sélectionne aucune face et active l'option 'Zoom plan'; si aucun objet de niveau de détail i+1 n'existe, l'utilisateur en est averti par message; si des objets de niveau de détail i+1 existent, ils sont proposés à l'utilisateur l'un après l'autre dans l'ordre de leur création jusqu'à ce qu'il valide l'affichage de l'un parmi eux

ou qu'il parvienne à la fin de la liste des objets. Cela est géré par la fenêtre de message et par l'affichage en vidéo inverse des faces associées au niveau de détail i+1 proposé (cf. figure 64).

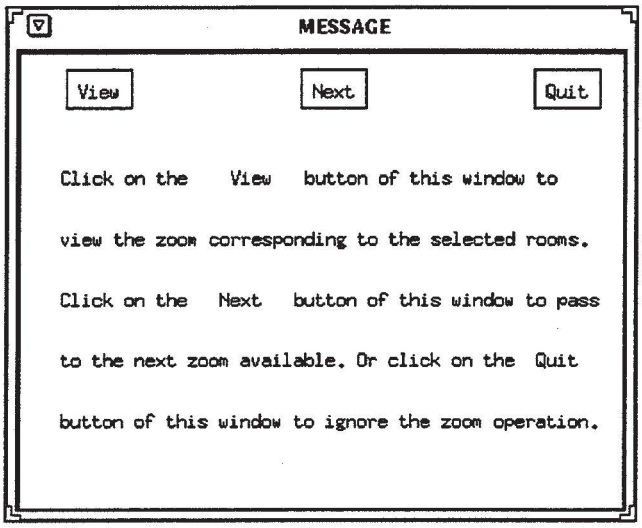
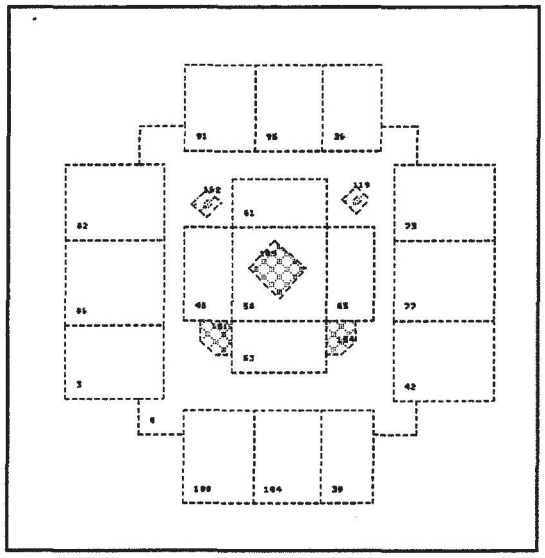


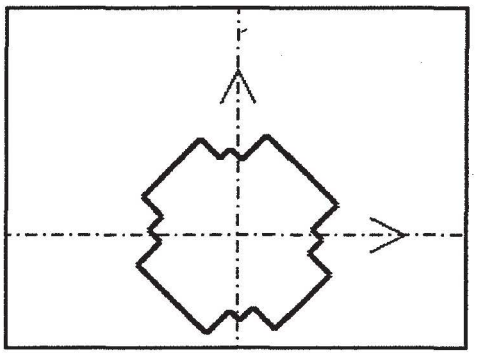
Figure 64 : message guidant le choix d'un chemin qui mène vers une des descriptions détaillées, pour un niveau de détail donné

d. Exemples d'utilisation du point de vue 'site' et de la hiérarchie dans les cartes planaires

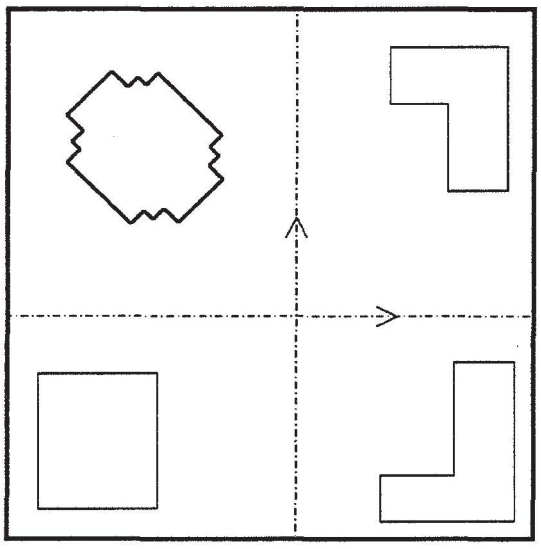
Un exemple de changement de point de vue du dessin bi-dimensionnel du niveau 0 d'un immeuble à sa représentation dans le plan 'site' est fourni dans la figure 65.



(a) carte planaire de bâtiment du niveau 0 d'un immeuble



(b) vue 'site' du même immeuble tourné d'un angle de 45 degrés



(c) vue d'un site constitué de plusieurs immeubles

Figure 65 : exemple de changement de point de vue

La figure 66 montre un exemple d'utilisation de la hiérarchie dans le modèle 2D$^{1/2}$ basé sur les cartes planaires pour une description du réseau et du bâtiment d'un étage d'immeuble. La figure 66(b) contient une description détaillée de la pièce 14 de la figure 66(a).

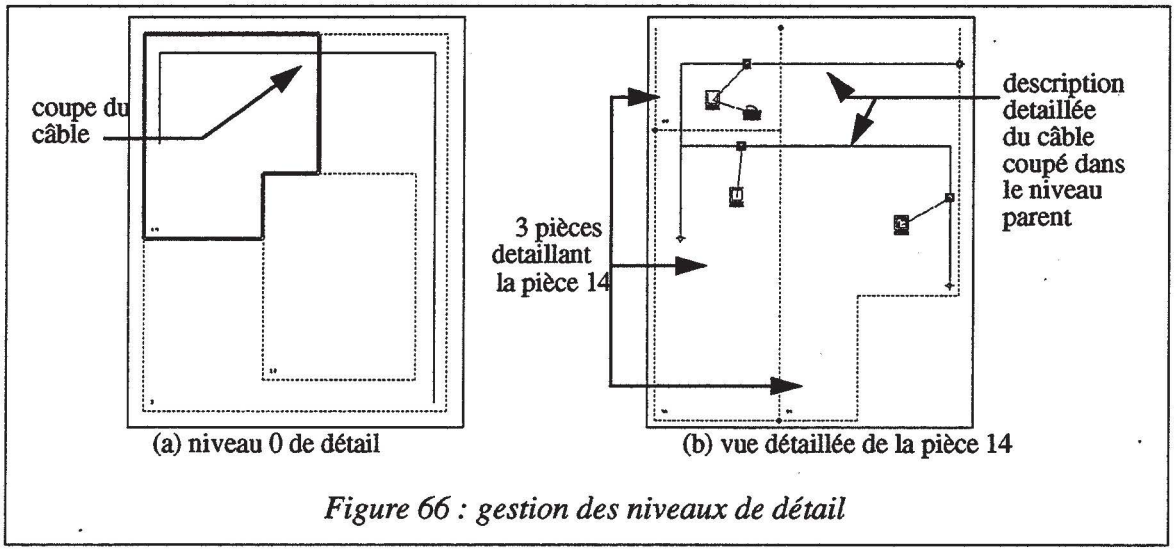


Figure 66 : gestion des niveaux de détail

2.5. Les objets sémantiques et les opérations

2.5.1. La création des objets

a. Les objets issus de la construction d'une carte planaire

Pour résumer, nous avons vu dans la section 2.4. de cette partie qu'à l'issue des constructions statique et incrémentale des cartes planaires, nous obtenons les objets sémantiques ci-dessous, chacun étant caractérisé par son identificateur :

- chaque arête portant une des textures de bâtiment est associée à un objet sémantique mur, faisant partie de la carte planaire de bâtiment. Cet objet possède, selon la texture de l'arête, soit l'attribut mur perforable, soit l'attribut mur non perforable. Ceci est fait par un champ supplémentaire de type entier dans la structure représentant les objets sémantiques. Ce champ intitulé type est une spécification de la sémantique,

- chaque face constituée d'une succession d'arêtes portant la texture de bâtiment est associée à un objet sémantique pièce par défaut, faisant partie de la carte planaire de bâtiment. L'attribut pièce peut être spécialisé ultérieurement par l'utilisateur en couloir, salle des machines informatiques ou pièce ordinaire,

- chaque arête portant une des textures de réseau est associée à un objet sémantique partie_de_câble, faisant partie de la carte planaire de réseau. Une succession non interrompue d'arêtes 'partie_de_câble' constitue un objet physique composé de toutes les parties (par agrégation). L'objet agrégat est appelé câble. Il possède, selon la texture des arêtes qui le composent, soit l'attribut câble épais, soit l'attribut câble fin, soit l'attribut câble ordinaire.

La structure de données *Objet* a été présentée en section B.3.2. du chapitre II de ce document. Cette structure possède parmi ses champs un champ union qui pointe sur la représentation graphique. Nous y avons rajouté un autre choix de type **struct liste_points** qui permet de représenter des gaines techniques horizontales installables le long de murs. Pour les objets mur et partie_de_câble, par exemple, le champ de représentation graphique pointe sur une structure d'arête. Pour l'objet câble, ce champ pointe sur nil. En effet, la représentation d'un câble est assurée par la représentation de toutes les parties de câble qui le constituent. La structure de connexion permet de lier un objet câble à tous les objets partie_de_câble qui le constituent, et permet ainsi sa représentation graphique.

b. Les objets icôniques

Nous rappelons ici que les objets icôniques ne peuvent être installés qu'en état de construction incrémentale du plan en cours de description. Les icônes peuvent représenter des installations de réseau (serveur, connecteur ...) ou des services inter-étages (sortie d'ascenseur ou d'escalier ou encore puits de gaine technique verticale). Les services inter-étages sont attribués au bâtiment tandis que les autres icônes sont attribuées naturellement au réseau. Les services inter-étages sont gérés par un seul bouton de l'interface de l'OMD placé dans la zone intitulée 'BUILDING ICONS'. Ce bouton est doté d'un menu à trois options (cf. figure 56(b)). Les icônes de réseau sont toutes accessibles depuis la zone de l'OMD intitulée 'NETWORK COMPONENTS' (cf. figure 56(b)). Cette zone est dotée d'un ascenseur permettant de parcourir l'ensemble des icônes rentrant dans le cadre de l'application.

c. Les objets texturés

Les objets texturés sont les faux-plafonds, les faux-planchers et les gaines techniques horizontales. Ces objets ne peuvent être installés dans un niveau d'étage que dans l'état de construction incrémentale des plans de l'étage.

Pour créer un faux-plafond (resp. un faux-plancher) dans une pièce, l'utilisateur doit **sélectionner la pièce** (cf. paragraphe de sélection), puis **activer le bouton** représentant le faux-plafond (resp. le faux-plancher). Les **boutons** représentant le faux-plafond et le faux-plancher sont dans la zone de F2D intitulée 'TEXTURES'. Un exemple montrant la manière de visualiser un faux-plafond a été fourni figure 57.

Pour créer correctement un objet gaine technique horizontale, l'utilisateur doit passer à la vue 'Building and network' (cf. figure 58), puis en l'état où aucun des boutons de sémantique n'est activé. A ce moment là, l'utilisateur voit apparaître un bouton concurrent de celui des murs et des câbles (cf. figure 56(b)) : le bouton 'H.shaft'. L'activation de ce bouton active celui de dessin ligne brisée. Une bonne installation de gaine technique horizontale demande à l'utilisateur de dessiner une ligne brisée qui longe des murs de l'étage courant. L'algorithme de recalage et la structure connexion de la figure 14 permettent d'établir un lien étroit entre la gaine technique horizontale installée et les murs qu'elle longe (cf. figure 67).

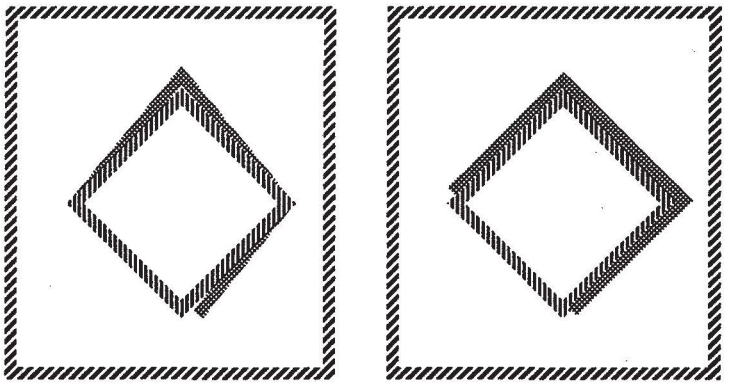
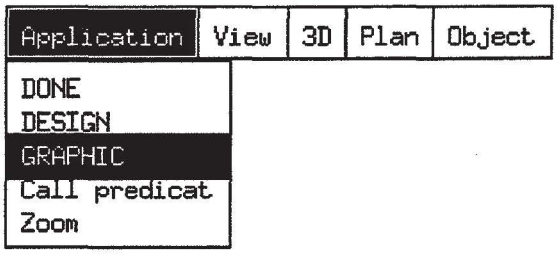


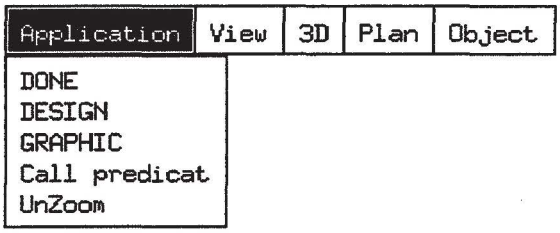
Figure 67 : création d'une gaine technique horizontale

2.5.2. La sélection/désélection des objets - les objets visualisés

Les objets peuvent être visualisés ou non en fonction de la vue courante dans la fenêtre F2D (bâtiment, réseau ou les deux à la fois) et de la position de l'ascenseur dans la fenêtre F3D. Un autre facteur concerne les objets icôniques intervenant lors de l'installation des icônes de terminateurs au bout des câbles. Ce facteur concerne aussi les icônes entrant en jeu lors de la mise en place d'un lien entre deux icônes de réseau (cf. paragraphe 2.5.3.). Il est relatif au choix de l'utilisateur : le mode 'Zoom' (resp. 'UnZoom') dans le menu associé au bouton de la barrette intitulé 'Application' (cf. figure 68).



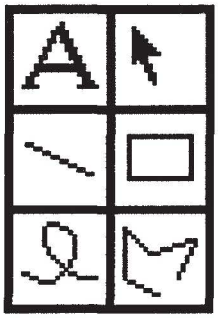
(a) mode UnZoom : les icônes représentant les connexions intermédiaires ne sont pas visualisées



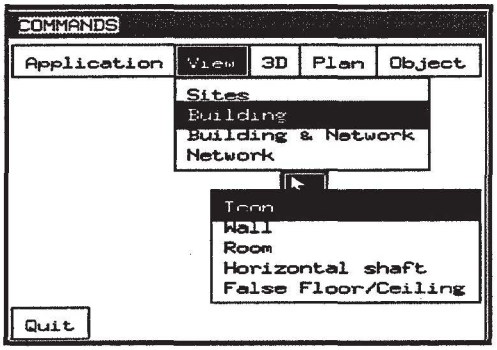
(b) mode Zoom : les icônes représentant les connexions intermédiaires sont visualisées

Figure 68 : le bouton 'Application' et les modes Zoom et UnZoom

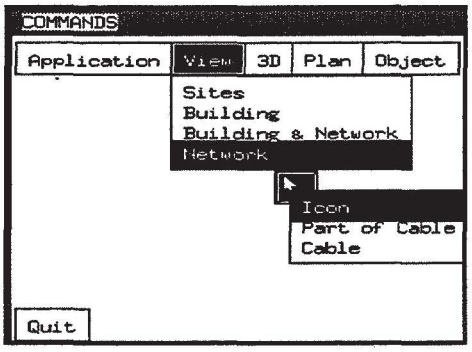
La sélection des objets ne peut se faire qu'après leur création donc forcément dans un état de construction incrémentale des plans auxquels ils appartiennent.



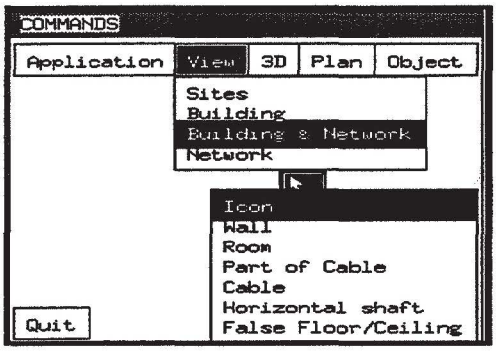
(a) le bouton 'flèche' de sélection et ses concurrents



(b) le menu de sélection en vue 'bâtiment seul'



(c) le menu de sélection en vue 'réseau seul'



(d) le menu de sélection en vue 'bâtiment et réseau'

Figure 69 : la fonctionnalité de sélection parmi les objets de bâtiment et de réseau d'un immeuble

Tout objet visualisé sur F2D ou F3D peut être sélectionné après activation du bouton de sélection. Le bouton de sélection est un bouton concurrent des boutons de dessin. Il n'est rendu visible à l'utilisateur qu'en état de construction incrémentale (cf. figure 69(a)).

Le **bouton flèche** de sélection est doté d'un **menu** dont les options dépendent de la vue courante (cf. figures 69(b), 69(c) et 69(d)). Dans la vue 'Site', le bouton de sélection n'a pas de menu associé. Avant de sélectionner un objet, l'utilisateur doit préciser sa nature en choisissant l'une des options du menu de sélection qui lui correspond, après avoir désigné le bouton de sélection. Le bouton de sélection et son menu fonctionnent de manière analogue aux boutons et au menu de dessin. Un objet dont on a indiqué la nature est sélectionné par une action adg à l'endroit où il se trouve.

Un objet sélectionné est visualisé en **vidéo inverse** ou en **gras** pour les segments. La resélection d'un objet sélectionné entraine sa **désélection**. Les **'objets' de type immeuble** dans la vue 'site' sont sélectionnables comme les autres.

2.5.3. La connexion/déconnexion des objets

Dans le domaine de la conception des réseaux, on a souvent recours à l'établissement de connexions entre les différents éléments qui composent un réseau. Connecter deux objets revient à créer un lien entre eux et la structure représentée figure 14 permet de modéliser les liens.

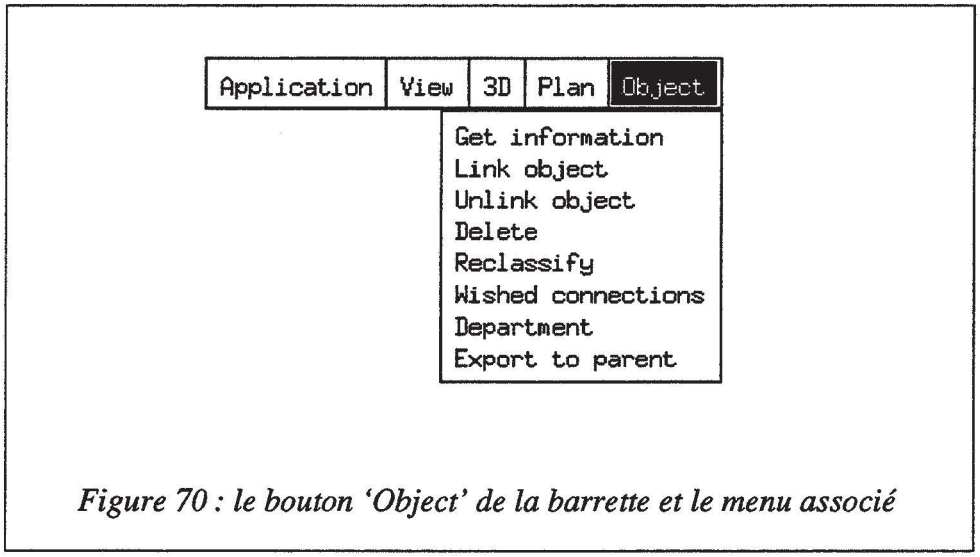


Figure 70 : le bouton 'Object' de la barrette et le menu associé

Il y a des connexions qui sont établies automatiquement : lors de l'installation d'une machine dans une pièce (relation dans), l'installation d'un terminateur en bout de câble (des éléments techniques additionnels s'ajoutent pour affiner la connexion entre le terminateur et le câble; se référer à la figure 72). A part ces connexions automatiques, des connexions entre objets icôniques peuvent être établies à l'aide de l'option 'Link object' du menu associé au bouton 'Object' de la barrette après sélection des deux objets à connecter (cf. figure 71). Certaines connexions

nécessitent une spécification technique de la part de l'utilisateur : l'information nécessaire a donc été placée au coeur de l'interface graphique, permettant ainsi de proposer à l'utilisateur un choix parmi les alternatives possibles à l'aide de la fenêtre de message (cf. figure 71).

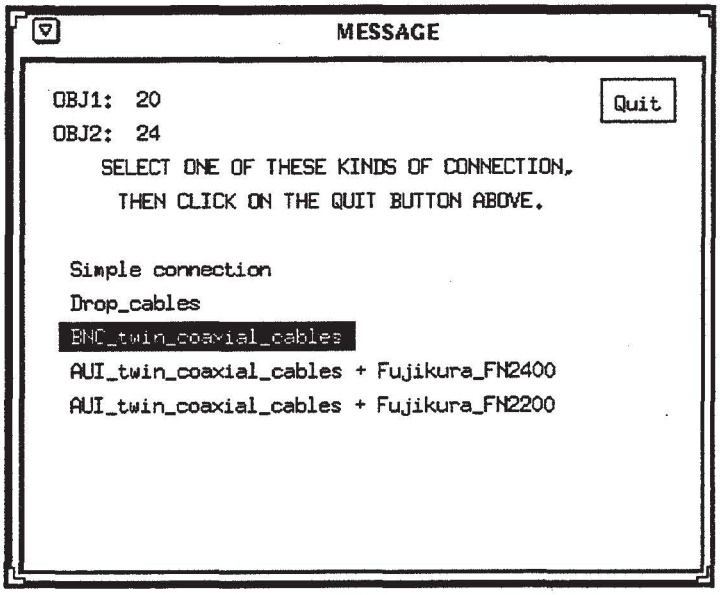


Figure 71 : message invitant l'utilisateur à choisir un type de lien

Pour pouvoir effectuer correctement les connexions, certaines procédures et fonctions ont été développées pour tolérer les imprécisions de l'utilisateur : projeter une icône de terminateur contre le bout de câble réseau le plus proche, recaler une installation de gaine technique horizontale pour la mettre contre les murs qu'elle est censée longer ...

La déconnexion entre objets icôniques peut survenir suite à deux types d'intervention de la part de l'utilisateur : soit l'utilisateur sélectionne deux objets à déconnecter puis choisit l'option de menu 'Unlink object' (cf. figure 70), soit il sélectionne un objet et le détruit par l'option de menu 'Delete' (cf. figure 70), auquel cas l'objet en question est déconnecté de tous ceux auxquels il était lié avant d'être détruit. La figure 72 ci-dessous montre différents objets icôniques impliqués dans des connexions, qui sont visibles ou non selon le mode courant 'Zoom' ou 'UnZoom' (cf. figure 68).

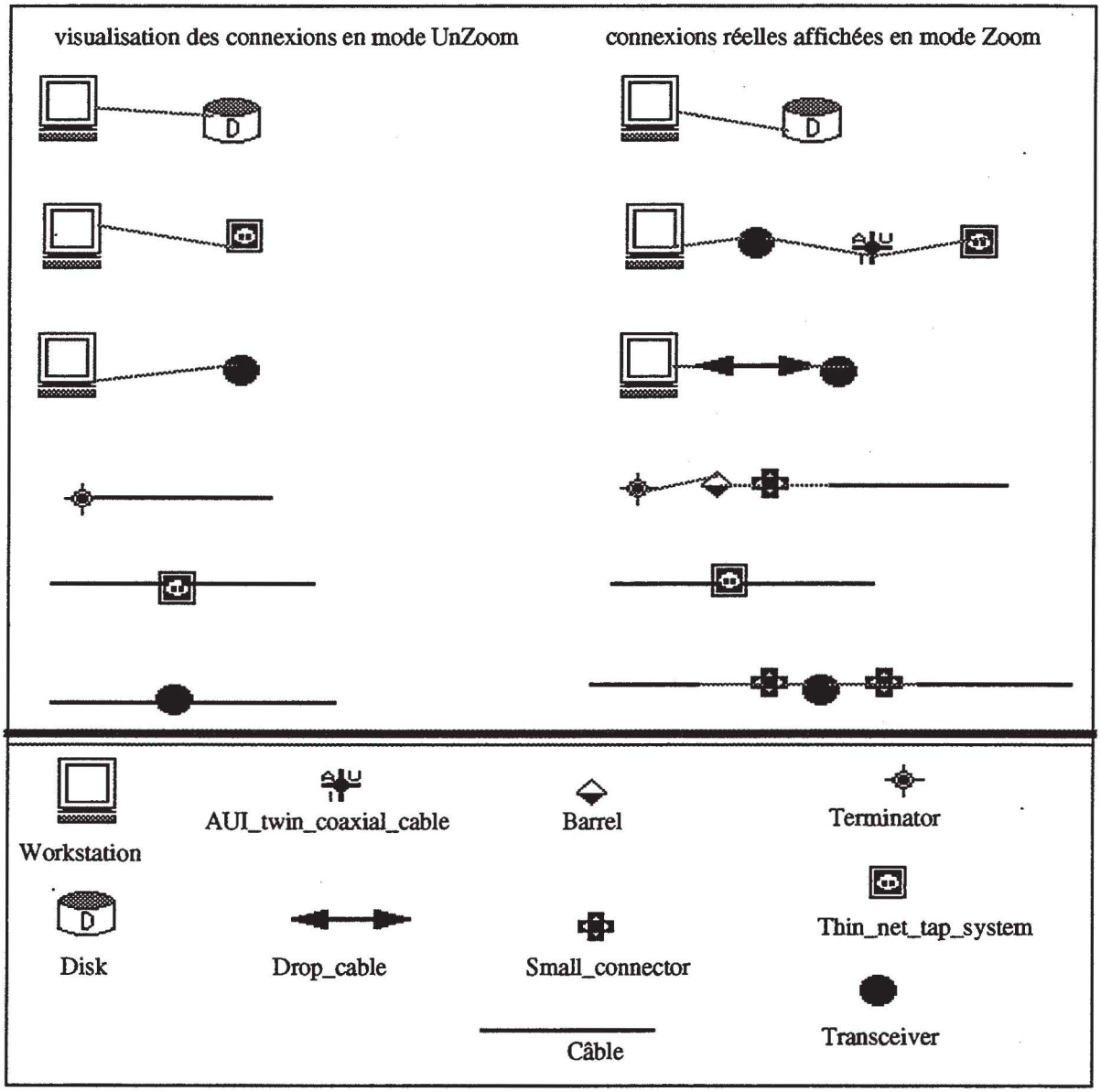


Figure 72 : les connexions (modes Zoom et UnZoom)

2.5.4. L'information sur les objets

L'utilisateur manipule un certain nombre d'objets graphiques à travers plusieurs modes et il doit parfois les désigner par leur identificateur. Cependant, à part les objets de type *Pièce*, les identificateurs des objets ne sont pas affichés à l'écran. Nous avons mis en place une option 'Get information' dans le menu associé au bouton 'Object' de la barrette (cf. figure 70). Le choix de cette option après sélection d'un objet permet de voir affiché l'identificateur de l'objet, sa classe, ainsi que les objets auxquels il est lié par des connexions. Nous avons distingué les connexions liant les

objets de même niveau de détail de celles intitulées 'détail', liant des objets qui appartiennent à des niveaux de détail différents. La fenêtre de message affiche, à chaque appel de la fonction d'information sur un objet, sa classe, son identificateur, le numéro d'étage auquel il appartient, le niveau de détail dans lequel il apparaît, les objets auxquels il est connecté avec la distinction 'zm-' pour les objets du niveau parent et 'zm+' pour les objets des niveaux fils immédiats. La figure 73 montre un exemple d'affichage des informations d'un objet sur la fenêtre de messsage : l'objet est une partie de mur (affiché en gras et délimité par des éléments de continuité de bâtiment), il est lié à des objets du niveau de détail parent (objet 17) et trois objets de niveaux de détail fils immédiat.

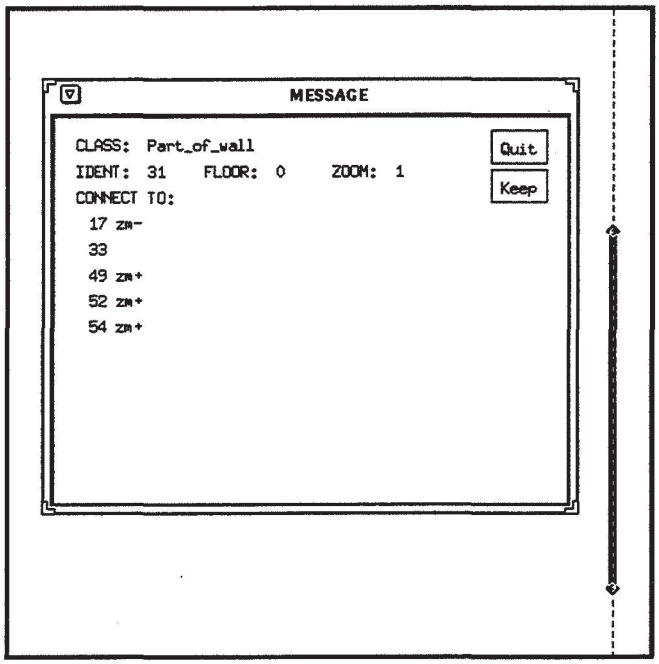


Figure 73 : l'affichage des informations d'un objet dans la fenêtre de message

La fonctionnalité d'information a été enrichie pour accéder aux informations d'un objet à partir de son identificateur; c'est à dire sans le sélectionner par désignation puisqu'on ne le connait pas (exemple : l'objet d'identificateur 33 de la figure 73). Pour cela, l'utilisateur choisit l'option de menu 'Get information' sans sélection préalable d'objet. L'interface lui propose de fournir l'identificateur de l'objet à désigner en le guidant par la fenêtre de message (cf. figure 63). Dans le cas de l'objet d'identificateur 33, l'interface nous informe que c'est l'objet de sémantique mur qui comprend la partie de mur sélectionnée. Nous rassurons le lecteur attentif qui aura remarqué que dans les options de menu de sélection, il n'y a pas de proposition de sélection pour les parties de mur, en lui disant que ici nous avons procédé de la manière inverse : d'abord une sélection du mur 33 qui par 'Get information' nous a livré ses constituants; ensuite 'Get information' par l'identificateur 31 nous a donné le message de la figure 73.

2.5.5. *La destruction des objets*

La destruction des objets est accessible après sélection des objets. La **sélection** d'un objet peut se faire soit par désignation directe soit par son identificateur via la commande 'Get information' suivie de 'Keep' dans la fenêtre de message (cf. figure 73). Le bouton 'Keep' de la fenêtre de message permet de garder sélectionné l'objet sur lequel l'utilisateur vient de s'informer.

La destruction des objets est gérée par l'option 'Delete' du menu associé au bouton 'Object' de la barrette (cf. figure 70). Après choix de cette option, un mécanisme de messages demande à l'utilisateur de confirmer sa demande ou l'avertit, si l'objet à détruire est une pièce par exemple, que l'exécution n'est possible que si l'utilisateur choisit de détruire tous les murs de la dite pièce.

L' autorisation de la destruction d'un objet entraine d'abord la rupture de toutes ses connexions avec les autres objets.

2.5.6. *Le déplacement des objets icôniques*

Le déplacement d'un objet icônique préalablement installé dans la zone de dessin peut être réalisé par l'utilisateur de manière assez simple : d'abord une action adg sur l'objet; puis une action adm; et enfin, une réinstallation de l'objet à l'aide d'une action adg.

2.5.7. *Le nombre de connexions des objets de type Pièce*

Parmi les connaissances nécessaires au système expert NEST pour la réalisation d'une conception de réseau figure le nombre de connexions désirées par l'utilisateur dans certaines pièces de son étage. L'affectation de cette connaissance pour les objets de type *Pièce* est gérée par l'option 'Wished connections' du menu associé au bouton 'Object' de la barrette et par un champ entier de la fenêtre de message. L'utilisateur doit activer l'option de menu 'Wished connections' après sélection de la pièce concernée.

2.5.8. *La reclassification des objets*

Les objets graphiques manipulés dans l'outil de manipulation directe étant tous modélisés en hiérarchies de classes dans l'expert sémantique, une partie de ces hiérarchies a été intégrée au coeur de l'OMD pour permettre à l'utilisateur une spécification ou une généralisation facile et rapide de ses objets (cf. figure 74).

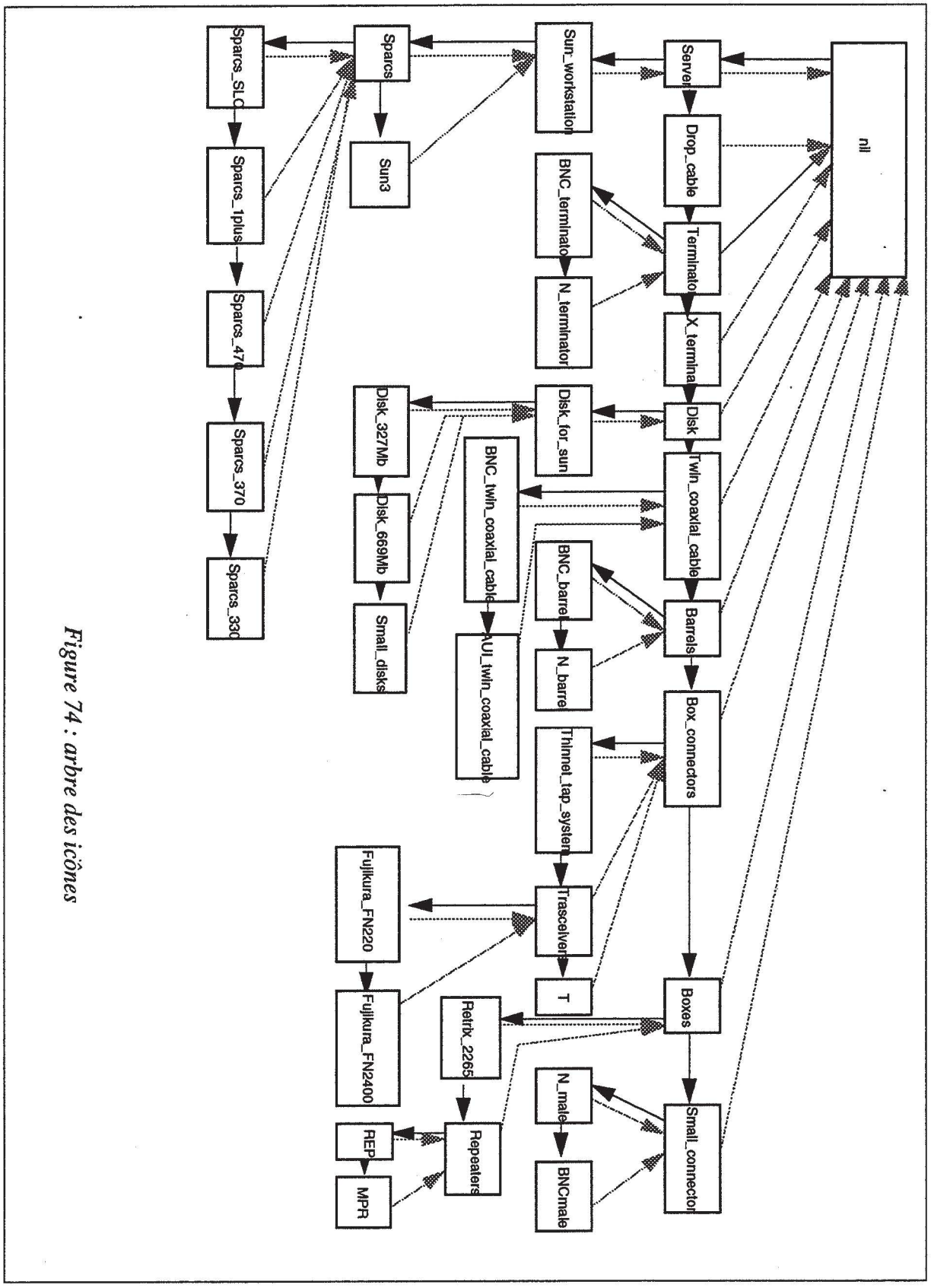


Figure 74 : arbre des icônes

La reclassification des objets est gérée par l'option 'Reclassify' du menu associé au bouton 'Object' de la barrette (cf. figure 70). L'utilisateur sélectionne d'abord l'objet à reclassifier puis active cette option de menu. L'opération de reclassification est guidée par la fenêtre de message qui propose à l'utilisateur de généraliser récursivement en désignant à chaque fois la classe parente, ou de spécialiser récursivement en désignant à chaque fois une des classes filles (cf. figure 75). L'objet courant change et le contenu de la fenêtre de message change avec lui jusqu'à ce que l'utilisateur choisisse d'arrêter le parcours de la hiérarchie par désignation du bouton 'Quit' de la fenêtre de message. L'objet sélectionné au départ devient alors une réplique du dernier objet courant dans le parcours de la hiérarchie. L'objet en tant que structure ne change pas. C'est seulement son champ 'représentation graphique' qui se retrouve modifié puisque la reclassification s'effectue au sein de grandes classes seulement : la classe des murs, la classe des câbles, la classe des icônes de réseau (cf. figure 75) ...

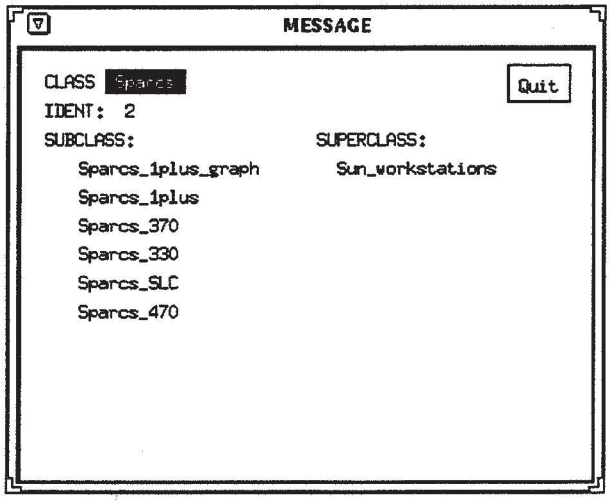


Figure 75 : reclassification d'une station de travail de type Sun

2.5.9. La départementalisation

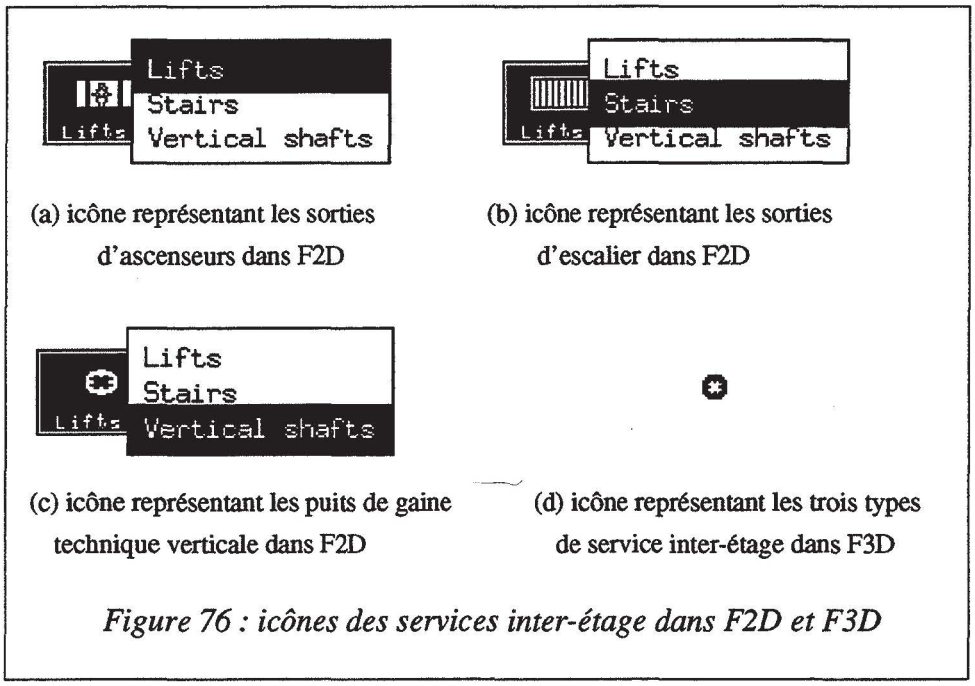
Une information supplémentaire utilisée par les experts en réseau est la départementalisation. En effet, un utilisateur peut concevoir lui même un réseau informatique et l'installer dans un bâtiment. L'application NEST est en mesure de lui fournir une analyse de la départementalisation. Pour cela, l'utilisateur doit lui fournir comme information les départements au sens 'réseau informatique'. L'entrée de cette information est gérée par l'option 'Department' du menu associé au bouton 'Object' de la barrette (cf. figure 70). L'utilisateur sélectionne d'abord des pièces de bâtiment puis active l'option 'Department' pour signifier au système qu'il les groupe dans un seul département au sens 'réseau informatique'.

2.5.10. Les opérations gérant les services inter-étages - '3D'

Les fonctionnalités inter-étages sont gérées par le menu associé au bouton de la barrette intitulé '3D' (cf. figure 59). Parmi les options de ce menu, nous avons traité la première option qui bascule de la commande 'Open 3D window' à la commande 'Close 3D winodw' selon l'état de F3D fenêtre dédiée à l'affichage tri-dimensionnel.

a. Alignement/Annulation de l'alignement des services inter-étages

Nous avons vu dans le paragraphe relatif à la création des objets icôniques que ces objets, a fortiori les services inter-étages, peuvent être installés depuis F2D ou F3D. Les services inter-étages sont des icônes qui représentent pour un étage soit la sortie d'une cage d'ascenseur, soit la sortie d'escaliers, soit un puits de gaine technique verticale (cf. figure 76).



(a) icône représentant les sorties d'ascenseurs dans F2D

(b) icône représentant les sorties d'escalier dans F2D

(c) icône représentant les puits de gaine technique verticale dans F2D

(d) icône représentant les trois types de service inter-étage dans F3D

Figure 76 : icônes des services inter-étage dans F2D et F3D

L'option 'Align Service Cable' du menu associé au bouton '3D' de la barrette permet d'aligner verticalement les icônes de service inter-étage. Après sélection d'icônes de service appartenant à des étages différents, l'activation de cette option permet d'aligner verticalement ces icônes sur la dernière sélectionnée d'entre elles. L'alignement est physique et s'accompagne d'une connexion qui propage le déplacement d'une icône à toutes celles qui sont alignées avec elle verticalement. C'est d'ailleurs l'une des raisons d'être de la fonctionnalité qui permet l'annulation de la connexion d'alignement vertical entre deux icônes et qui est gérée par l'option 'Delete vertical alignement' du menu associé au bouton '3D' de la barrette.

b. Installation/Destruction d'un câble d'épine dorsale verticale joignant deux icônes de service inter-étage

Les besoins de l'application nous ont amené à modéliser les objets *câble d'épine dorsale verticale* joignant les réseaux installés sur plusieurs étages. Ce sont des objets appartenant à l'immeuble tout entier et pas particulièrement rattachés à un étage. Nous avons choisi de les inclure parmi les données de réseau de l'étage de niveau 0. La mise à jour de ces objets est très délicate. En effet, chaque objet de type *câble d'épine dorsale verticale* groupe par transitivité toutes les icônes liées entre elles par une connexion créée par l'opération : sélection de deux icônes de service inter-étage puis choix de l'option 'Put Backbone Cable' du menu associé au bouton '3D'. Lors de l'affichage, un bout de câble vertical matérialise par un trait épais la connexion liant deux services inter-étage et marquée 'BACKBONE' (cf. figure 60). L'annulation d'une connexion de ce type est gérée par l'option 'Delete Backbone Cable' du menu associé au bouton '3D'. Une sélection de deux icônes de service inter-étage suivie de l'activation de cette option de menu entraine l'annulation de la connexion 'BACKBONE' entre les deux icônes.

c. Déplacement/Redimensionnement des 'objets' niveau d'étage

Nous précisons ici que seules les vues globales des niveaux d'étage sont visualisées dans la fenêtre F3D (niveau de détail 0). Or, dans sa description bi-dimensionnelle dans F2D, l'utilisateur ne peut pas savoir exactement les dimensions des étages au dessus et en dessous. Donc, nous avons proposé des fonctionnalités tri-dimensionnelles qui permettent d'avoir un repère commun à tous les étages : les boîtes transformées tri-dimensionnelles de la boîte rectangulaire englobant tous les tracés de plan de l'immeuble. L'utilisateur peut ainsi déplacer les dessins d'un étage dans son plan horizontal sans le sortir de sa boîte tri-dimensionnelle. De même, un utilisateur peut redimensionner les plans d'un étage toujours sans le sortir de sa boîte tri-dimensionnelle. Le déplacement est géré par l'option 'Move Storey' du menu associé au bouton '3D'; le redimensionnement est géré par l'option 'Resize Storey' du menu associé au bouton '3D'. Ces deux fonctionnalités fonctionnent de la manière suivante : d'abord une activation de la fonctionnalité voulue, puis la fenêtre de message s'affiche en indiquant à l'utilisateur comment faire pour l'exécuter sur un étage à l'aide de la touche de milieu de la souris.

2.5.11. L'exportation d'un objet au niveau de détail parent

C'est une fonctionnalité très importante dans notre conception de la hiérarchie (cf. section 6 de la partie C du chapitre II). Des objets décrits dans un niveau de détail assez élevé peuvent être importants pour des fonctionnalités associées à une vue globale (niveau de détail faible). Cette fonctionnalité permet une duplication dans le père de la représentation d'un objet dans un de ses fils, avec une mise à jour adéquate des liens 'détail' introduits par la hiérarchie parmi les objets. Cette fonctionnalité est gérée par l'option 'Export to parent' du menu associé au bouton 'Object' de la barrette. La sélection d'un objet suivie de l'activation de cette option permet l'exportation de l'objet vers le niveau de détail père, si père il y a.

2.5.12. La traduction des objets - communication avec les autres modes de l'interface MMI²

Les principales commandes assurant la communication avec les autres modes de l'interface MMI² sont groupées dans le menu associé au bouton intitulé 'Application' de la barrette (cf. figure 68). Trois options de ce menu gérent deux commandes chacune : les modes DESIGN/ EDIT, les modes GRAPHIC/GESTUEL et les modes Zoom/UnZoom.

La première option du menu associé au bouton 'Application' est une commande qui permet de faire la traduction, pour l'application et les autres modes de dialogue, de toutes les données saisies graphiquement. Cette commande n'a de sens qu'en mode DESIGN : l'utilisateur prend le soin de bien finaliser son dessin graphique avant de le communiquer à l'application pour une analyse par exemple. Le mode EDIT est l'opposé du mode DESIGN : chaque interaction graphique est immédiatement traduite pour l'application et le restant du système. L'explication des modes Zoom/UnZoom a été donnée dans la section 2.5.3. de cette partie. La commande 'Call predicat' permet à des utilisateurs experts d'interroger l'application par utilisation de prédicats Prolog.

Les modes GRAPHIC/GESTUEL permettent l'orientation des interactions de l'utilisateur soit vers le graphique (outil de manipulation directe) soit vers le gestuel (reconnaissance de commandes gestuelles).

Nous appelons traduction des objets l'opération de conversion des données emmagasinées dans la structure de données carte planaire en clauses Prolog. Ces clauses Prolog sont transférées au gestionnaire graphique qui les traduit en expressions CMR (langage de représentation interne commun à tous les modes de MMI²) qui seront transférées vers le contrôleur de dialogue.

Chaque type d'objet dans la structure de données de l'outil de manipulation directe est décrit au sein du gestionnaire graphique par une classe ayant des méthodes et des attributs. Chaque classe a une méthode de traduction associée permettant d'instancier tous les éléments nécessaires pour la décrire. La traduction dans son ensemble est gérée par les deux prédicats Prolog suivants :

putobjetcs/3 : prédicat d'arité 3 permettant de créer une instance d'une classe donnée

putobjectvalue/3 : prédicat d'arité 3 permettant de modifier la valeur d'un attribut d'un objet donné

3. COMMUNICATION ENTRE L'OMD, LE GESTIONNAIRE GRAPHIQUE ET LES AUTRES MODES DE MMI^2

Comme le montrent les figures 41 et 43 qui situent l'OMD au sein de l'architecture générale de MMI^2, le gestionnaire graphique assure un niveau intermédiaire de représentation interne. Il transforme les informations reçues de l'OMD en expressions CMR. Inversement, il transforme les expressions CMR fournies par les différents experts du système en prédicats Prolog qui permettent d'activer des méthodes (procédures) de l'OMD.

La communication entre l'OMD et le gestionnaire graphique est présentée ici selon deux aspects : en mode entrée, permettant à l'utilisateur de communiquer avec le système; en mode sortie, permettant au système de présenter les informations à l'utilisateur. Ensuite, nous fournissons un exemple qui permet de voir le flux d'informations au sein de l'interface MMI^2 auquel participe activement le gestionnaire graphique.

3.1. Mode entrée

En mode entrée, l'OMD envoie les informations, en direction du gestionnaire graphique, sous forme de prédicats Prolog. Nous avons présenté brièvement dans la section 2.5.12. l'idée directrice de la traduction pour le système de l'ensemble des objets formant les plans saisis. Une utilisation normale de l'OMD est d'abord une approche globale de la topographie du bâtiment dans lequel sera installé le réseau (mode DESIGN), suivie d'une série d'interactions avec l'application NEST (mode EDIT). Le passage en mode EDIT (cf. section 2.5.12.) fait que chaque opération accomplie par l'utilisateur produit un prédicat informant le système et lui demandant l'autorisation d'exécuter cette opération. Si l'autorisation est refusée, l'OMD informe l'utilisateur qu'il lui est impossible d'exécuter cette opération. Un exemple de prédicat utilisé en entrée est celui de création dont la description suit :

gr_net_u_create/3 (gr_net pour dire mode graphique de NEST et u pour utilisateur, donc mode entrée)

arg1 : identificateur de l'objet créé
arg2 : type de l'objet ou classe (par exemple Thinnet_tap_system)
arg3 : variable d'autorisation

Par ce prédicat, l'OMD informe le système que l'utilisateur veut créer un objet. Le type de l'objet est défini dans arg2. Si le système autorise la création de l'objet, l'argument 3 reçoit la valeur OK; sinon il reçoit la valeur ERROR et l'objet ne sera pas inséré dans la liste des objets de l'OMD.

3.2. Mode sortie

Le gestionnaire graphique dialogue avec l'OMD en utilisant des prédicats Prolog. Ces prédicats activent des méthodes agissant sur les structures de l'OMD et des procédures d'une bibliothèque graphique conçue au dessus de celle qu'offre le gestionnaire des fenêtres. Ces procédures, ainsi que toutes les méthodes de mise en oeuvre des structures de l'OMD sont

écrites en langage C. Un exemple de fichier minimal permettant la transition Prolog - C est le suivant :

extern ([a_C_func],['fichier.o']).

gr_prédicat(liste_variable) :-
a_C_func (liste_variable).

"a_C_func" est une procédure écrite en C et qui doit être déclarée préalablement comme étant une fonction externe. Un exemple de prédicat utilisé en mode sortie est le suivant :

gr_net_s_unlink/2 (s pour système et donc mode sortie)

arg1 : identificateur d'un objet
arg2 : identificateur d'un objet

Ce prédicat permet au système de détruire la connexion entre les deux objets désignés par leurs identificateurs.

3.3. Exemple montrant la coopération du graphique avec les autres modes de MMI^2

Considérons le cas d'une commande de déplacement d'un objet formulée par l'utilisateur via le langage de commande de l'interface MMI^2. Cette commande fait participer l'expert graphique en mode sortie (affichage). Elle génére tout un flux d'informations décrit ci-dessous et représenté dans la figure 77.

1- Utilisateur : DEPLACE Machine0 Pièce3.

2- Le module du langage de commande vérifie la validité syntaxique de la commande.

3- Le module du langage de commande crée une expression CMR et vérifie la sémantique de la commande : il sélectionne dans son lexique le prédicat CMR qui représente la commande "DEPLACE". Il demande à l'expert sémantique si les arguments existent dans l'arbre sémantique. Après quoi, il demande à l'expert du domaine (qui consultera la base de connaissances) le type des arguments (dans ce cas la variable Machine0 est de type SUN3/60 et Pièce3 est de type LOCAL). Il demande à l'expert sémantique si les types des arguments vérifient les paramètres de la commande. L'expert sémantique donne son accord et ainsi le module du langage de commande crée une expression CMR pour la commande DEPLACE.

4- L'expression CMR est envoyée à l'expert de l'interface qui la renvoie au contrôleur de dialogue. Le contrôleur de dialogue envoie une copie à l'expert du contexte de dialogue et à l'expert du modèle utilisateur, interprète l'expression CMR et l'envoie à l'expert du domaine.

5- L'expert du domaine exécute le prédicat DEPLACE(Machine0,Pièce3) qui déplace, dans la base de connaissances, la Machine0 dans Pièce3.

6- L'expert de domaine demande au module graphique de mettre à jour l'affichage en déplaçant la machine (Machine0) dans la pièce (Pièce3).

7- L'expert du domaine informe le contrôleur de dialogue. Celui-ci informe l'utilisateur que la commande a été exécutée en créant une expression CMR qui est envoyée à un module de la langue naturelle (le français).

8- L'expression est passée au module de la langue naturelle qui la transforme en français.

9- Le texte est alors envoyé à l'expert de l'interface qui l'affiche sur l'écran.

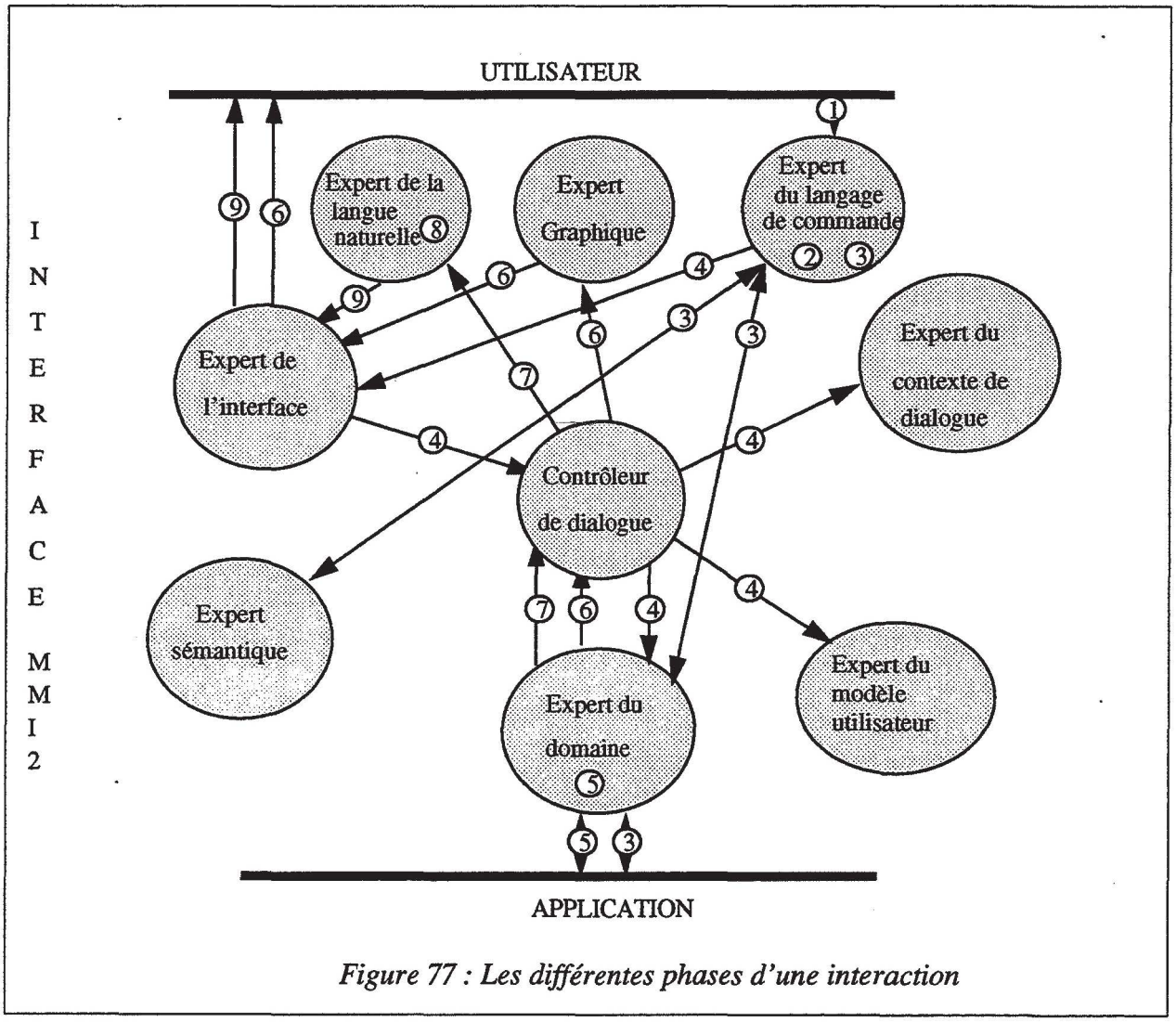


Figure 77 : Les différentes phases d'une interaction

CHAPITRE IV

L'ÉVALUATION DE L'INTERFACE GRAPHIQUE

A. INTRODUCTION

La multiplication des logiciels et la valorisation des techniques nouvelles ont donné une part importante aux Interfaces Homme-Machine (IHMs) dans le choix des logiciels. Une recherche permanente de la qualité dans les IHMs s'est instaurée et a fait émerger plusieurs techniques pour leur évaluation, ces évaluations se basant aussi bien sur des aspects techniques (vitesse, erreurs ...) que sur des aspects ergonomiques.

Ce que l'on se propose d'évaluer, c'est l'outil de manipulation directe qui a été réalisé à l'EMSE pendant les trois premières années du projet MMI^2. Le développement de cet outil s'était accompagné de tests portant sur ses fonctionnalités. L'EMSE a ensuite participé à l'effort d'intégration de l'outil graphique dans le prototype de MMI^2 exposé à Bruxelles en Novembre 1991. L'outil graphique a donc été, au même titre que les autres modes de dialogue de MMI^2, évalué par le scénario proposé par RAL [WILS 91].

Notre objectif est d'évaluer l'outil graphique développé au sein de l'EMSE afin de compléter l'évaluation du prototype de MMI^2 effectuée à l'aide du scénario proposé par RAL. Par cette évaluation, nous voudrions sonder la satisfaction des utilisateurs par rapport à l'interface en général. Nous voudrions également tester l'utilité des différents outils de dessin mis à la disposition des utilisateurs : main levée, segment, segment élastique (ou ligne brisée) et rectangle. Nous désirerions aussi avoir une idée sur la complexité de l'interface graphique et sur la durée moyenne que nécessite son apprentissage. Enfin, nous voudrions confronter l'interface à certains critères ergonomiques.

Dans ce chapitre, on trouvera d'abord une étude bibliographique des nombreuses publications portant sur l'évaluation. Ensuite, notre technique d'évaluation sera exposée. Puis, nous rapporterons les valeurs numériques des facteurs mesurés lors de cette évaluation et nous analyserons ces résultats numériques tout en les confrontant aux problèmes ergonomiques relevés.

B. LES INTERFACES HOMME-MACHINE ET L'ÉVALUATION

1. INTRODUCTION

La littérature portant sur l'évaluation des IHMs est abondante et ceci n'a pas forcément pour effet de simplifier la tâche à accomplir par les novices que nous sommes dans le domaine de l'évaluation. En effet, la tâche d'évaluation peut être vue sous différents angles. On a d'abord une classification selon la phase de développement de l'IHM pendant laquelle intervient l'évaluation. Ensuite, on distingue les évaluations selon les objectifs qui ont guidé la réalisation de ces évaluations. Enfin, on différencie les évaluations suivant les techniques utilisées pour les effectuer. Ces techniques peuvent être basées sur des données objectives ou subjectives, de surface ou de profondeur.

Dans [HOWA 87], une étude taxonomique des techniques d'évaluation des IHMs est exposée. Cet article contient une description de plusieurs facteurs qui interagissent dans une évaluation.

Parmi ces facteurs figurent les objectifs de l'évaluation dont la spécification est, et doit être, le point de départ de l'évaluation. En effet, les objectifs doivent définir clairement l'IHM à évaluer. Ce qui suppose que l'on situe, dès la spécification des objectifs, la tâche d'évaluation par rapport au cycle normal de développement d'un système informatique. Ensuite, connaissant les objectifs de l'évaluation et l'IHM à évaluer, le choix parmi les techniques existantes ou l'élaboration d'une technique nouvelle est arbitré par les ressources et les moyens disponibles pour accomplir la tâche d'évaluation. Un moyen simple pour évaluer une IHM est de recourir aux experts dans le domaine des IHMs. C'est une technique d'évaluation qui peut être faite rapidement et intervenir à tout moment dans le cycle conception-développement de l'IHM. Elle peut consister en l'invitation d'un expert à évaluer les performances d'un système par rapport aux recommandations ou standard existants. Outre cette technique, que l'on appelle l'évaluation experte, des évaluations d'IHMs peuvent être conduites par des concepteurs en utilisant :

- des guides de recommandations,
- des modèles théoriques/formels,
- des utilisateurs.

Ce sont ces trois techniques que nous allons présenter dans les paragraphes suivants.

2. L' ÉVALUATION DES IHMs À L'AIDE DES GUIDES DE RECOMMANDATIONS

Certaines recommandations ont un support empirique ou théorique; d'autres, au contraire, résultent de l'expérience dans les dialogues Homme-Ordinateur et les facteurs humains. Bien sûr, parmi ces recommandations, certaines sont liées à des technologies d'interfaces révolues; d'autres permettent au concepteur de choisir dans un ensemble limité d'alternatives issues des facteurs humains et lui évitent de se perdre dans des considérations infinies.

Le fait de suivre les recommandations dès le début de la conception guide le concepteur dans une direction justifiée dès le départ et évite une remise en cause trop coûteuse du système final. Faire un choix parmi toutes les recettes proposées n'est pas facile et le recours à un expert en la matière, même si on doit le payer en tant que consultant externe, est une sage décision. En effet, développer une mauvaise interface et essayer de corriger ce qui ne va pas en elle une fois qu'elle est terminée coûte nettement plus cher que de s'occuper de la qualité de l'interface dès le début de la conception.

Une manière de faire l'évaluation des IHMs est de spécifier les objectifs de qualité que l'on désire atteindre en termes de critères. Cette spécification doit se faire dans la phase de spécification de l'IHM. L'évaluation de l'IHM par rapport à ces critères peut alors se faire avant, pendant ou après le cycle de conception-développement de celle-ci.

[SHNE 87] propose cinq critères mesurables pour l'évaluation d'IHMs. Ces critères sont :

1- le temps moyen d'apprentissage,
2- la vitesse d'exécution des tâches,
3- le pourcentage d'erreurs commises,
4- la satisfaction subjective de l'utilisateur,
5- la retention humaine des commandes à travers le temps.

Il fournit également un exemple de test d'acceptabilité :

" 30 utilisateurs seront entraînés à utiliser le système pendant 45 minutes. On leur donnera à faire pendant 15 minutes l'ensemble de tâches ci-joint. Le pourcentage moyen d'accomplissement des tâches doit être supérieur à 80% et le nombre moyen d'erreurs doit être inférieur à 3".

Une autre approche consiste à évaluer l'IHM par rapport à des critères ergonomiques. Une vue sommaire sur ces techniques a été présentée pour la première fois dans le cadre du projet MMI^2 dans [DARS 91]. Une étude a été menée à l'INRIA [BAST 91] pour valider les définitions de 18 critères ergonomiques.

Dans ce qui suit, nous présentons les aspects positifs et négatifs des guides de recommandations.

2.1. Obstacles à l'utilisation des guides

L'idée de suivre les guides de recommandations constitue un problème qu'il est difficile de trancher : GIBBONS a rapporté dans [GIBB 92] plusieurs recueils destinés à être utilisés par les concepteurs, les ingénieurs et les professionnels des facteurs humains. Ces recueils portent l'étiquette guide de recommandations de conception (design guidelines) [BROW 88], [RIVL 90], [SCAP 87], [SMIT 86], l'étiquette manuel (handbook) [HELA 88], [SALV 87] ou encore l'étiquette standard [ANSI 88], [DIN 87].

Nielsen et Molish rapportent dans [NIEL 90] que l'ouvrage [SMIT 86], qui est sans doute le plus souvent référencé, compte environ mille règles à respecter. Ce volume important est intimidant pour les concepteurs [BAST 91], et est de nature à les pousser à conduire des évaluations heuristiques selon le sens commun et leur propre intuition. Par contre les ergonomes, qui ont un certain degré d'expertise du domaine, sont en mesure de bien interpréter les recommandations [SCAP 90], et ont une certaine facilité à utiliser les guides.

La difficulté vient bien sûr de la multiplicité des guides et de la diversité dans la manière de les structurer. [BAST 91] porte l'attention sur le fait qu'un critère donné n'est pas nécessairement défini de la même façon par tous les auteurs qui l'utilisent et signale qu'il arrive que des critères différents fassent référence au même contenu, et inversement, que des critères identiques soient définis par des contenus différents. GIBBONS décrit dans [GIBB 92] l'effort de compilation de plusieurs guides dans le but d'en extraire une information experte pour l'évaluation des systèmes à menus. D'une part, il déplore que SHNEIDERMAN consacre dans [SHNE 87] un chapitre entier aux systèmes à menus, sans prendre le soin de donner une définition claire des systèmes à menus. D'autre part, GIBBONS donne des exemples de multiples références au même objet de conception : "coded entry = keyded codes = option designators = selection number = sequential number coding ; option phrasing = option wording ; ..."

[POLS 92] donne l'exemple d'une recommandation difficile à appliquer : "minimiser la charge de mémorisation". En effet, cette recommandation ne dit pas comment mesurer la charge de mémorisation et ne propose aucune méthode pour la minimiser.

Pour éviter de se perdre dans les généralités, Molish et Nielsen ont choisi de réduire les recommandations utilisées dans leurs évaluations à 9 principes d'utilisabilité [MOLI 90]. Cet article rapporte une expérience d'identification des problèmes d'utilisabilité d'une IHM particulière par 77 concepteurs et programmeurs. La pertinence des réponses apportées par les évaluateurs est jugée par 3 personnes spécialisées dans le dialogue Homme-Ordinateur. Les auteurs ont découvert que l'ensemble des évaluateurs n'est pas arrivé à lister tous les problèmes d'utilisabilité reconnus par les spécialistes. Inversement, les évaluateurs ont soulevé de réels problèmes d'utilisabilité auxquels les spécialistes que sont les auteurs n'avaient pas pensé. Ces mêmes auteurs ont trouvé dans [NIEL 90] que leurs critères peuvent servir à conduire des évaluations heuristiques d'IHMs par

des non spécialistes, à condition d'utiliser une technique d'agrégation (des groupes de 3 à 5 évaluateurs) qu'ils exposent par ailleurs. Cependant, [POLS 92] signale que ces 9 principes d'utilisabilité ne sont pas adéquats pour détecter dans un système téléphonique des problèmes sérieux qui dépendent des buts de l'utilisateur et de la manière dont ils évoluent.

2.2. Une bonne utilisation des guides de recommandations

Ces controverses autour de l'utilité des guides de recommandations étant rapportées, on ne doit pas oublier de signaler que selon [BAST 91], Smith et Mosier ont montré que leur guide [SMIT 86] s'avère utile : les personnes à qui ils l'ont envoyé l'ont lu, l'ont utilisé, envisagent de l'utiliser de nouveau et de le recommander à d'autres personnes. Les auteurs constatent aussi que les recommandations qu'il contient sont surtout utilisées au début de la conception bien qu'elles soient aussi utiles pour l'évaluation des systèmes existants ou en cours de conception.

Des études récentes telles que [GIBB 92] se sont servies du guide [SMIT 86], comme d'autres guides [SCAP 87] et [WILL 84], pour arrêter la définition de certains critères et leur associer des règles expertes (397 règles "simples") destinées à l'évaluation d'interfaces utilisateurs à base de menus. Un exemple de ces règles est le suivant :

"Menu dialogues should be selected if the future users will be inexperienced".

Cette règle est relative à l'adéquation du dialogue à base de menus. Elle a l'avantage d'être utile dès la phase de conception des IHMs. Elle permet de favoriser le dialogue à travers des menus par rapport au dialogue à l'aide d'un langage de commandes dans le cas où l'IHM est destinée à des utilisateurs inexpérimentés.

On peut également trouver des recommandations dans des documents ne portant pas les étiquettes : guide de recommandations, manuel ou standard. En effet, les expériences d'évaluations à thèmes divers se multiplient. On trouve des évaluations comparatives d'IHMs prouvant le caractère discriminant d'une technique d'évaluation [PLAI 92], [CHIN 88], ainsi que des évaluations d'objets de dialogue d'une IHM telles que l'évaluation des raccourcis clavier de commandes [WALK 88], l'évaluation des différentes façons de concevoir des menus [WALK 91] ... Dans ce dernier article par exemple, on recommande à l'issue de l'étude d'utiliser des bordures dans les menus afin d'éviter à l'utilisateur d'être précis et de lui permettre d'être plus performant au regard du temps de sélection. On propose également de laisser perméable une surface à gauche du menu pour prendre en compte le cas où l'utilisateur souhaite sortir du menu sans choix d'option.

3. MODÈLES THÉORIQUES/FORMELS POUR L' ÉVALUATION DES IHMs

Les évaluations qui s'appuient sur des modèles théoriques et/ou formels interviennent assez tôt dans le cycle de conception-développement des IHMs. Elles permettent de prédire la complexité d'un système et par conséquent les performances des utilisateurs. Cinq prédictions émanant de la théorie de Hayes et Broadbents [HAYE 88] en sont un parfait exemple. Cette théorie compare les interfaces à commandes et les interfaces à manipulation directe, et prédit que les utilisateurs d'une interface à commandes doivent :

(1) faire moins d'erreurs,
(2) utiliser plus de temps pour résoudre un problème donné,
(3) être négativement affectés par une réduction de la saillance de l'IHM
(la saillance d'une interface pouvant être la quantité et le type de retour d'informations qu'elle offre),
(4) être plus aptes à dire ce qu'ils ont appris,
(5) préférer cependant les interfaces à manipulation directe aux interfaces à commandes.

[BAST 91] a classifié ces modèles en modèles de tâches, modèles linguistiques et modèles cognitifs de l'interaction. Selon [HOWA 87], l'évaluation basée sur les modèles théoriques/ formels nécessite une expertise considérable et les représentations de l'utilisateur et du système doivent y être détaillées en fonction des situations.

3.1. Les modèles de tâches

Parmi ces modèles, on peut citer GOMS (Goals, Operators, Methods, Selection rules) [CARD 83] et KLM (Keystroke-Level Model) de [CARD 80]. Ces modèles permettent de prédire les performances de l'utilisateur d'une IHM à partir des spécifications de sa conception.

Une définition française de GOMS a été donnée dans [SENA 90] : "GOMS décrit les performances de l'utilisateur d'un système donné en termes de buts poursuivis, d'opérateurs (actions élémentaires mises en jeu pour satisfaire le but poursuivi), de méthodes (définissant les procédures permettant d'atteindre le but) et de règles de sélection des méthodes".

GOMS prédit la durée d'exécution d'une tâche donnée en la décomposant en buts et sous-buts. A un niveau de détail donné, des données contextuelles et un temps de préparation mentale (choix d'une méthode ou d'un opérateur) peuvent accroître le temps de réalisation d'une tâche.

KLM est un cas particulier du modèle GOMS où la décomposition des tâches descend jusqu'au niveau des actions physiques réalisées sur le clavier. [SENA 90] et [HART 90] précisent que ce modèle prédit le temps d'exécution d'une **tâche routinière** par un **utilisateur expérimenté** ne commettant **pas d'erreurs**, ce qui constitue de fait une limite de ce modèle en ce qui concerne

la prédiction des performances des utilisateurs. Un autre inconvénient est la longue durée nécessaire à la modélisation des tâches en KLM due au niveau de détail assez élevé.

Un des avantages de ces modèles est qu'ils permettent d'attribuer une valeur correspondant à l'effort requis par un "expert standard" pour accomplir un ensemble de tâches prédéfinies à l'aide d'un dispositif. Si ces modèles ne constituent pas un outil d'évaluation à part entière, ils permettent de structurer les tâches pour des utilisations diverses. [GONG 90] a utilisé GOMS pour produire un manuel minimal qui servirait en priorité à des utilisateurs novices ayant des buts précis. Une autre utilisation de GOMS serait de l'étendre pour fournir les procédures de recouvrement associées à chacun des états d'erreur identifiés, comme proposé par [CARD 83].

3.2. Les modèles linguistiques

Les modèles linguistiques tentent de rendre explicite, sous forme d'une grammaire, la structure de l'interface. Un de ces modèles, ALG (Action Language Grammar) [REIS 83], construit un modèle des actions nécessaires à l'accomplissement d'une tâche. Le processus de réalisation de la tâche est décrit sous forme d'une grammaire dont les règles établissent les correspondances entre les buts et les actions élémentaires (procédures) nécessaires à leur atteinte.

[BAST 91] rapporte que :

- le nombre d'actions élémentaires différentes pour atteindre un but,
- la longueur des séquences d'actions pour une tâche donnée,
- le nombre de règles non nécessaires,
- le nombre de règles pour les séquences terminales similaires

sont des métriques utilisées pour évaluer les IHMs à l'aide du modèle ALG.

CLG (Command Language Grammar) [MORA 81] est un autre modèle linguistique. Il consiste en une décomposition hiérarchique de l'interface. Ce modèle est surtout utilisé pour la conception des IHMs en les décrivant en termes de tâche, de sémantique, de syntaxe et d'interaction. Il marque une avancée par rapport au modèle ALG en intégrant dans la description un composant physique lié à l'environnement matériel (composants matériels et unités d'Entrée/ Sortie). CLG peut également être utilisé pour l'évaluation [SHAR 87]. Mais, son principal inconvénient, d'après [SENA 90], est qu'il ne peut être utilisé pour évaluer une IHM développée avec une méthodologie autre que CLG.

3.3. Les modèles cognitifs de l'interaction

Selon [BAST 91], les modèles cognitifs, et plus particulièrement CCT (Cognitive Complexity Theory) [KIER 85], permettent une simulation de l'interaction entre l'utilisateur et le dispositif à partir de trois éléments : un modèle de l'utilisateur exprimé sous forme de règles de production; une représentation de la tâche utilisant la notation du modèle GOMS; et un modèle de

l'interface. Dans CCT, l'interface est décrite à l'aide du formalisme GTN (Generalized Transitions Networks) qui permet des représentations hiérarchiques à différents niveaux d'abstraction.

[SENA 90] cite quelques indicateurs retenus pour caractériser la complexité. Ce sont :

- le nombre total de règles de production requises pour modéliser la tâche,
- le nombre de règles de production declenchées,
- le nombre de conditions et d'actions d'une règle de production,
- les piles de buts maintenues en mémoire de travail,
- le nombre maximal de buts en mémoire pour réaliser une fonction donnée.

Un autre modèle cognitif basé également sur la simulation est PUM (Programmable User Model). [YOUN 89] le définit comme étant une architecture contrainte par la cognition qui peut être programmée par le concepteur.

Ce modèle propose de dire aux concepteurs :

" *If you think your design is so great, then program this architecture to behave like a user using your interface to perform some task. If you find that a straightforward and satisfactory thing to do, you may take it as suggestive evidence that the interface will indeed be easy to use. But if you find that some aspect of it gives you trouble, take that as evidence that people will have difficulty with the same aspect*' [YOUN 89].

L'utilisation de PUM donne aux concepteurs des retours d'information à deux moments : premièrement, pendant qu'ils sont engagés dans la programmation du PUM; deuxièmement, quand le modèle est réellement mis en fonctionnement pour en déduire des prédictions.

D'après [YOUN 89], PUM doit avoir recours aux techniques d'Intelligence Artificielle pour réagir comme un moteur qui applique la connaissance aux situations pour générer un comportement. L'idéal serait que PUM fasse automatiquement ce que toute personne qualifie d' "évident" dans le contexte, et qui, par conséquent, n'a pas besoin d'être dit.

Un modèle concurrent de PUM est celui exposé dans [POLS 92] : le parcours cognitif. L'idée de ce modèle cognitif consiste en une simulation manuelle du comportement de l'utilisateur. Il s'intéresse surtout à l'anticipation des problèmes d'apprentissage des fonctionnalités des IHMs.

De ce point de vue, le parcours cognitif permet l'évaluation des IHMs assez tôt dans le cycle de conception d'un système.

Cela demande au concepteur de préparer et d'exposer devant un groupe de rapporteurs les éléments suivants :

- une description détaillée de l'interface,
- le scénario des tâches,
- des hypothèses explicites sur la population des utilisateurs et sur le contexte d'utilisation,
- les séquences d'actions avec lesquelles un utilisateur peut accomplir une tâche avec succès à l'aide de l'interface à évaluer.

Les rapporteurs doivent examiner de près les tâches sélectionnées par le concepteur. Ces tâches doivent correspondre à des fonctionnalités que doit posséder l'interface. A ce niveau, on adopte le point de vue d'une personne qui utilise l'interface pour la première fois; et toute supposition sur l'état du système quand l'utilisateur commence le travail doit être notée.

Dans un parcours des actions associées à une tâche, ce groupe doit considérer le comportement de l'interface et ses effets sur l'utilisateur dans le but d'identifier les actions qui seraient difficiles à choisir ou à exécuter par le membre "moyen" de la prétendue population d'utilisateurs. A chaque fois qu'un des rapporteurs déclare qu'une action peut causer des difficultés, il doit fournir des arguments théoriques, des données empiriques ou des arguments relevant de l'expérience et du sens commun. De même pour les actions qui ne devraient pas poser de problème.

Le modèle utilise une hiérarchie des buts baptisée : structure "and-then" similaire aux hiérarchies postulées par le modèle GOMS. La structure "and-then" et ses avantages sont présentés en détail dans [POLS 92].

La technique du parcours cognitif est avantageuse par rapport à PUM parce qu'elle dispense les concepteurs de produire une simulation exécutable. En effet, les moyens de l'Intelligence Artificielle ne sont généralement pas utilisés par les équipes de conception de logiciels. Quant à la constitution du groupe de rapporteurs, [JEFF 91] a montré qu'un groupe constitué d'ingénieurs du logiciel peut procéder à une évaluation même si leur potentiel pour la détection des problèmes est nettement inférieur à celui des professionnels des interfaces utilisateurs (35 problèmes d'utilisabilité détectés dans une interface graphique, comparés à 105 problèmes détectés par une évaluation heuristique conduite par des professionnels des interfaces utilisateurs).

4. LES TECHNIQUES D' ÉVALUATION FAISANT APPEL AUX UTILISATEURS

Les techniques d'évaluation faisant appel aux utilisateurs, comparées aux techniques précédemment exposées, interviennent assez tard dans le cycle de conception-développement d'un système informatique. Elles opèrent de préférence sur des prototypes [PLAI 92], mais aussi sur des produits finis [MACK 89].

Dans les évaluations faisant appel aux utilisateurs, des données objectives (taux d'erreurs ...) aussi bien que des données subjectives (degré de satisfaction ...) peuvent être recueillies. [HOWA 87] propose quatre niveaux de perception chez un utilisateur qui peuvent être étudiés lors d'une évaluation d'IHM. Ce sont les niveaux physiologique, comportemental, cognitif et affectif. Les auteurs rapportent également dans cet article les techniques qui ont été le plus souvent utilisées pour les aborder.

Le recueil ou l'enregistrement des données dans ce type d'évaluations s'effectue à l'aide d'instruments dont les effets sur le degré de confiance à leur accorder sont plus ou moins importants. Un facteur qui peut aussi modérer les conclusions des évaluations est le choix de la population d'utilisateurs. De plus, la disponibilité des utilisateurs et l'acceptation de la technique qui leur est proposée, sont autant de facteurs qui peuvent compromettre ces évaluations. [BAST 91] donne comme exemple de méthode qui pourrait être rejetée par les utilisateurs, celle de l'enregistrement automatique des actions sur les différents dispositifs de commandes (mouchards électroniques).

Des travaux portant sur l'évaluation proposent des techniques assez variées. Certains utilisent l'enregistrement vidéo d'utilisateurs pendant leur interaction avec l'IHM à évaluer : [ANTI 90] et [LUND 85]. Cette technique est en général couplée avec celle qui consiste à demander aux évaluateurs de penser à haute voix et d'enregistrer leurs pensées pendant l'interaction.

Dans [ANTI 90], cette technique a été utilisée pour évaluer des systèmes d'aide au déplacement par voiture. Les sujets devaient lire à haute voix tout panneau de route et toute indication de direction qui seraient utilisés pour choisir un itinéraire; ceci devait être utilisé pour aider à la classification des coups d'oeil en dehors du véhicule enregistrés par vidéo. Cette évaluation utilise deux caméras : une fixée sur la tête du conducteur pour enregistrer la vue devant lui et une autre disposée spécialement pour suivre le mouvement des yeux du conducteur. Les données recueillies par ces caméras sont objectives et relèvent des niveaux physiologique et comportemental de perception des utilisateurs.

D'autres évaluations se sont servies d'un matériel encore plus sophistiqué : un traqueur à lumière infrarouge et un autre magnétique pour suivre le mouvement des yeux pendant que la tête est en mouvement [HEND 89].

Certaines évaluations s'intéressent aux niveaux cognitif et affectif de la perception des utilisateurs : en dernière étape des expériences menées dans [HEND 89], les sujets devaient ordonner les interfaces qu'ils ont manipulées par ordre de préférence.

Parmi les moyens qu'on peut utiliser pour collecter des informations d'ordre subjectif, on trouve les questionnaires [SVEN 91]. Les utilisateurs sont souvent amenés à y répondre en attribuant une note à l'IHM vis à vis du thème sur lequel porte la question. [CHIN 88] fournit une version validée (c'est la cinquième version) d'un questionnaire sondant la satisfaction d'utilisateurs d'IHMs : QUIS (5).

Dans les expériences décrites dans [GONG 90], les utilisateurs devaient donner leur avis sur certaines caractéristiques du manuel, en donnant à chaque caractéristique une note entre 1 et 5 : 1 pour une caractéristique pas utile du tout et 5 pour une caractéristique très utile.

[SENA 90] expose les cinq étapes de la méthodologie élaborée dans le projet Esprit HUFIT consacré à la mise au point d'outils facilitant l'intégration des facteurs humains dans la conception des systèmes d'information [NOVA 87a], [NOVA 87b]. C'est la dernière étape de cette méthode qui s'occupe de collecter des données subjectives : elle consiste à remplir un questionnaire d'utilisabilité dans lequel les sujets doivent donner une note de facilité d'usage, une note de facilité d'apprentissage et une note d'utilité de chaque caractéristique. Une note globale est fournie avec les suggestions d'amélioration.

[SHNE 87] propose deux versions de questionnaire destinées à l'évaluation des IHMs par des utilisateurs : une version courte et une version longue.

[SENA 90] expose les avantages et les inconvénients du système des questionnaires et la manière de les utiliser.

[BAST 91] a relevé quelques points critiques des évaluations faisant appel aux utilisateurs :

1- le temps consacré à l'observation des utilisateurs est généralement trop long, comme l'est celui nécessaire au codage ou décodage des données recueillies, à leur analyse et finalement à leur interprétation.

2- certaines méthodes font appel à des appareils coûteux et difficiles à mettre en oeuvre.

Un autre point négatif de ces méthodes selon [POLS 92] est qu'elles ne fournissent pas de diagnostic aux problèmes identifiés contrairement à la technique du parcours cognitif, sans oublier le besoin de localiser et de travailler avec des utilisateurs représentatifs.

C. L' ÉVALUATION DE L'INTERFACE GRAPHIQUE DE MMI^2

Pour effectuer cette évaluation, nous avous choisi de suivre le plan d'évaluation recommandé dans [HOWA 87] :

- spécification des objectifs de l'évaluation ;
- recherche de l'information nécessaire pour atteindre ces objectifs ;
- choix des techniques d'évaluation capables de fournir cette information ;
- collecte et analyse des résultats ;
- décisions de recommandations pour revoir la conception de l'interface ou pour réaliser des évaluations complémentaires.

1. OBJECTIFS ET ENVIRONNEMENT DE L' ÉVALUATION

Rappelons les objectifs que nous avons fixés à cette évaluation. Nous voudrions sonder la satisfaction des utilisateurs par rapport à l'interface en général. Nous voudrions également tester l'utilité des différents outils de dessin (main levée, segment, segment élastique, rectangle) mis à la disposition des utilisateurs. Nous désirerions aussi avoir une idée sur la complexité de l'interface graphique et sur la durée moyenne que nécessite son apprentissage. Enfin, nous voudrions confronter l'interface à certains critères ergonomiques. Ce travail d'évaluation de l'outil graphique développé au sein de l'EMSE sera en fait un complément de l'évaluation du prototype de MMI^2 effectuée à l'aide du scénario proposé par RAL.

Les objectifs que nous nous sommes fixés s'expriment donc en termes d'appréciation et de facilité d'apprentissage de l'interface graphique par des utilisateurs représentatifs. La technique d'évaluation faisant appel aux utilisateurs semblait s'imposer, et le fait de disposer d'un prototype nous a encouragé à nous orienter vers cette direction. Une investigation faite dans ce sens nous a permis de nous assurer de l'appui du personnel du département informatique de l'EMSE. Ainsi, nous pouvions compter sur la disponibilité de dix volontaires pour la réalisation d'une telle évaluation. Ces volontaires ont une expérience plus ou moins confirmée dans l'utilisation des interfaces graphiques et/ou des logiciels de synthèse d'images. Le groupe qu'ils forment est constitué d'une secrétaire, de sept chercheurs et de deux ingénieurs.

Un certain nombre de remarques doivent être faites :

- l'interface MMI^2 incluant tous les modes offre des moyens de dialogue complexes : nombre de fenêtres, possibilité de réaliser une même tâche de plusieurs façons différentes ... Cette complexité risquait de distraire les expérimentateurs et de les détourner de l'objet à évaluer.

- un utilisateur de l'outil graphique intégré avec le restant des modes de MMI^2, observe un ralentissement du temps de réaction du système une fois passé en mode EDIT. Ce qui est normal, car à partir de ce moment là, le système MMI^2 traduit toutes les actions de l'utilisateur en expressions CMR (Common Meaning Representation), contrairement au mode DESIGN.

- nous voulons évaluer l'outil graphique et non la globalité du système MMI^2. Il faut noter qu'une étude portant sur l'évaluation de l'interaction entre les différents modes du système MMI^2 a été réalisée dans le laboratoire de LEEDS.

A partir de ces remarques, nous avons trouvé naturel que les évaluateurs travaillent dans un environnement où l'outil graphique développé au sein de l'EMSE est détaché du reste du système MMI^2.

2. TECHNIQUES CHOISIES POUR EFFECTUER L' ÉVALUATION

Pour sonder la satisfaction des utilisateurs vis à vis de l'outil graphique, nous avons choisi d'utiliser un questionnaire, en prenant comme base la version validée dans [CHIN 88] que nous avons traduite en français et adaptée aux particularités de notre outil.

L'approche choisie pour atteindre les autres objectifs comporte deux étapes :

- la première a pour objectif de connaître l'aptitude des utilisateurs à maîtriser et à utiliser efficacement un des outils de dessin integrés dans l'interface, et à tester la préférence des utilisateurs.

- la deuxième étape est conçue pour offrir à chaque utilisateur plus de temps que celui passé dans la première étape, afin de lui permettre d'apprendre à utiliser l'interface à son rythme, et de nous permettre par la suite d'évaluer la complexité de l'outil graphique.

Une évaluation qui consisterait seulement en la mesure d'un certain nombre de critères est vide de sens dans notre conception des choses. En effet, la motivation des volontaires est plus grande quand leur participation a des effets appréciables; une amélioration de la qualité de l'interface suite à l'évaluation serait le meilleur aboutissement de leurs efforts. N'étant pas experts

dans le domaine du dialogue homme-machine et n'ayant pas les moyens de nous assurer les services d'un expert, nous avons pensé faire participer les évaluateurs dans le travail de détection des problèmes d'utilisabilité que présente l'interface. Les résultats du questionnaire et des deux étapes de l'évaluation seront alors complétés par une liste des problèmes d'utilisabilité détectés par les évaluateurs. Nous avons choisi d'effectuer une analyse modeste et non fastidieuse de ces problèmes d'utilisabilité en vue d'une classification. Notre choix s'est porté sur [BAST 91] qui contient la définition d'un nombre raisonnable de critères ergonomiques (18 critères) validés et exprimés de façon claire en langue française.

Ainsi, nous n'avons pas eu beaucoup d'hésitations à faire un choix parmi les techniques d'évaluation existantes. Notre volonté d'explorer le domaine de l'évaluation des IHMs conjuguée à l'obstacle du coût nous a conduit à écarter l'évaluation experte. L'interface graphique étant à un stade très avancé dans son cycle de développement, nous avons estimé qu'une évaluation à l'aide des modèles théoriques/formels serait coûteuse. D'autant plus que sa conception ne s'est pas faite à l'aide d'une modélisation explicite des tâches.

L'étude bibliographique effectuée nous a été très utile en raison de notre inexpérience dans le domaine de l'évaluation des IHMs. Elle nous a permis d'acquérir un minimum de culture nécessaire dans le travail d'observation des évaluateurs pendant l'évaluation. Dans notre cas, l'observation devait se faire de manière naturelle c'est à dire sans enregistrement sonore ou vidéo. Ces techniques ont été écartées car leur utilisation nécessite non seulement un matériel sophistiqué mais aussi une compétence considérable dans son maniement. Sans cette compétence, l'évaluation ne se ferait pas dans un climat de sérénité.

Dans les paragraphes qui suivent, nous allons décrire les deux étapes de l'évaluation de l'interface graphique ainsi que la planification de l'évaluation dans son ensemble. La participation des évaluateurs étant récompensée par une élimination progressive des problèmes d'utilisabilité dans le cadre des développements futurs. Il est important de noter que l'outil de manipulation directe décrit dans le chapitre précédent tient compte des résultats de cette évaluation.

2.1. Description de la première étape de l'évaluation : expérience 1

Dans cette expérience, les évaluateurs devaient réaliser le dessin de la figure 78 à l'aide des outils de dessin intégrés à l'interface graphique dans les conditions suivantes :

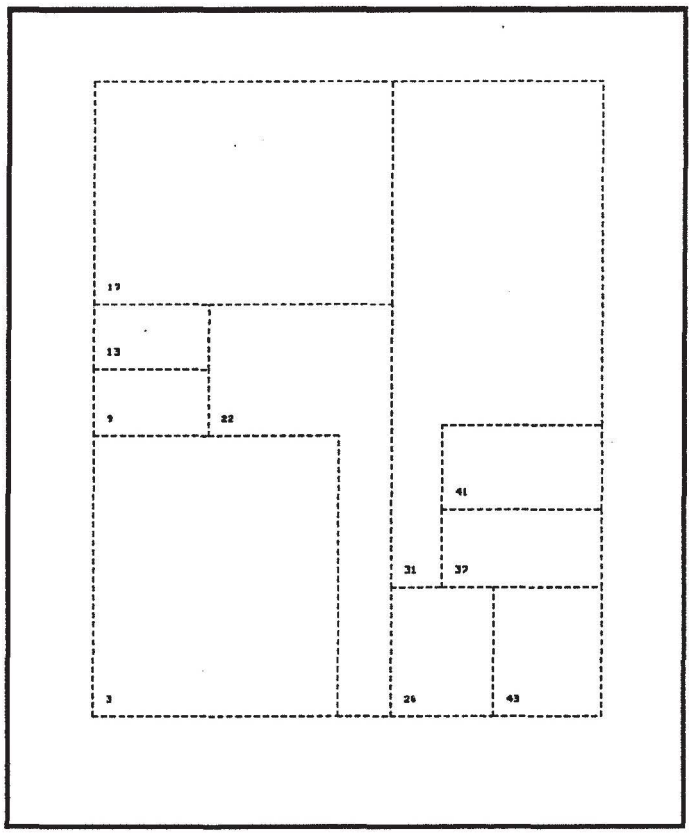


Figure 78 : dessin à réaliser pendant l'expérience 1

- on place l'évaluateur face à l'interface graphique et on lui demande de remplir la partie du questionnaire (cf. annexe A) traitant de la satisfaction visuelle (nous avons essayé de ne pas être un facteur perturbant durant l'exécution de cette tâche).

- on donne à l'évaluateur le manuel d'utilisation de l'interface graphique, ainsi que l'énoncé de l'expérience. Il sait donc ce qu'il doit réaliser par la suite.

- pour aller rapidement à l'essentiel de cette expérience, l'évaluateur est tenu de faire une lecture sélective du manuel. En effet, dans sa lecture, l'évaluateur doit prendre en compte le fait que le dessin à réaliser sera le dessin d'un bâtiment constitué d'un seul étage. Après (ou pendant) cette lecture, l'évaluateur doit se familiariser avec les outils de dessin et la démarche à suivre pour mettre son dessin à main levée au propre à l'aide de la notion de carte planaire supportée par l'interface. Cette phase "d'échauffement" est fixée à 20 minutes maximum.

- le dessin à réaliser (cf. figure 78) est ensuite fourni à l'évaluateur. Il sait qu'il sera chronométré et arrêté dans sa tâche s'il dépasse la durée fixée de cinq minutes.

Le dessin devait être réalisé cinq fois par chacun des évaluateurs avec utilisation :

- uniquement de l'outil main levée,
- uniquement de l'outil segment,
- uniquement de l'outil segment élastique (ligne brisée),
- uniquement de l'outil rectangle,
- des différents outils selon les préférences de l'utilisateur. Nous avons suivi les évaluateurs dans leur travail pour savoir quel est l'outil qu'ils utilisent le plus fréquemment dans cette expérience. Nous leur avons demandé ensuite de nous indiquer l'outil de dessin qu'ils préférent parmi les quatre testés.

- on note la durée de l'expérience,

- on aide l'évaluateur à remplir la première donnée (qui correspond au temps qu'a duré cette expérience pour lui) dans son livret d'apprentissage (cf. annexe C).

- on note les problèmes d'utilisabilité que l'évaluateur soulève.

Nous nous sommes tenu à côté des évaluateurs pour répondre à leurs questions, et pour recueillir leurs impressions et suggestions.

2.2. Description de la deuxième étape de l'évaluation : expérience 2

Cette expérience, intitulée apprentissage de l'outil, commence au lancement de l'expérience 1. L'évaluateur garde le manuel d'utilisation de l'outil graphique après avoir participé à l'expérience 1. Il a été tenu de noter sur son livret d'apprentissage le temps qu'il consacre à l'apprentissage de l'outil graphique : lecture du manuel d'utilisation, essai des différentes composantes de l'outil ou simple contemplation de l'interface sur écran ou sur copies d'écran. De même, nous nous sommes mis à la disposition des évaluateurs pour donner les explications nécessaires concernant l'utilisation de l'outil et pour recueillir les problèmes d'utilisabilité qu'ils signalent. Chacun des expérimentateurs peut décider par lui-même d'arrêter sa formation quand il juge qu'il a appris à bien se servir de l'outil. Le livret d'apprentissage nous permet de savoir le temps qu'il a fallu à chaque expérimentateur pour se former, le nombre de jours sur lesquels s'est étalée sa formation ... Nous avons incité les évaluateurs à faire de leur mieux pour que la période d'apprentissage ne dépasse pas trois semaines. Mais, comme le lecteur pourra le constater dans la section relative aux résultats, nous avons enregistré beaucoup d'exceptions dues aux occupations multiples des évaluateurs juste avant les vacances d'été.

Après l'apprentissage, chaque évaluateur a été soumis à un test (cf. annexe B) qui consiste en un certain nombre de questions portant sur un dessin fini, réalisé en utilisant l'interface et affiché sur l'écran d'une machine. Ensuite, l'évaluateur devait apporter quelques modifications à ce dessin. Nous avons compté sur la discrétion des évaluateurs, et nous leur avons demandé de ne pas

divulguer le contenu du test.

Ce test a été subi et noté deux fois par chaque évaluateur : la première fois juste après sa décision de fin de formation, et la seconde après au moins une semaine de coupure complète avec l'outil. Pendant les deux réalisations du test, les évaluateurs avaient accès au manuel d'utilisation de l'outil graphique.

Le livret d'apprentissage a été fourni à chaque évaluateur au début de l'expérience 1. L'évaluateur devait rendre le questionnaire dûment rempli à la fin de l'expérience 2.

2.3. Planification de la tâche de l'évaluateur

Avant de commencer les expériences d'évaluation, différents documents ont été produits :

1- Le questionnaire portant sur la satisfaction des évaluateurs vis à vis de l'outil. Chaque question attribue à l'outil une note échelonnée entre 0 et 9. Ces deux nombres sont associés à deux qualificatifs opposés, clairement indiqués dans chaque question. Ces deux qualificatifs opposés correspondent respectivement au pire et au meilleur des cas.

2- Le manuel d'utilisation de l'outil. Le travail d'évaluation est pour nous une occasion pour recueillir des remarques et suggestions destinées à améliorer la version primitive du manuel d'utilisation déjà produit.

3- Le test subi par les évaluateurs à l'issue de leur apprentissage de l'outil.

4- Le dessin à réaliser par les évaluateurs au cours de l'expérience 1 traitant de l'utilité des outils de dessin.

5- Un document aidant les évaluateurs à enregistrer le temps consacré à l'apprentissage. C'est ce que nous avons appelé le "Livret d'apprentissage" dans ce qui précède.

3. RÉSULTATS DE L' ÉVALUATION DE L'INTERFACE GRAPHIQUE DE MMI^2

Dans cette section, nous analyserons de façon globale les résultats numériques qui sont regroupés dans l'annexe D, et qui sont de deux sortes :

- des valeurs mesurées ou observées,
- des réponses fournies par les évaluateurs à des questions précises (scores).

Nous donnerons ensuite le résultat de la collecte des problèmes d'utilisabilité auprès des utilisateurs.

3.1. Résultats de l'expérience 1

Comme on peut le constater en regardant le tableau 1 (cf. annexe D), la moitié des évaluateurs ne connaissait pas le système de fenêtrage Suntools. Ce système de fenêtrage est, en effet, celui choisi pour supporter l'interface évaluée. De plus, deux parmi les évaluateurs (les sujets 5 et 7) n'attribuent pas un préjugé favorable aux interfaces à manipulation directe. Et comme le tableau 1 l'indique, ces sujets là ont réussi leur première expérience en n'enregistrant aucun échec.

Le tableau 1, comme les autres tableaux d'ailleurs, a été organisé de manière à ce qu'une lecture directe des performances des évaluateurs puisse être faite facilement. Cependant, nous désirons porter l'attention du lecteur sur certains faits. D'abord, l'utilité des outils rectangle et segment de droite est clairement démontrée dans ce tableau. Nous avons noté que dans la réalisation libre du dessin (cf. figure 78), les évaluateurs ont généralement utilisé l'outil rectangle pour tracer la boîte rectangulaire où s'inscrit le dessin à réaliser de toute évidence. Ensuite, dans leur majorité, ils ont utilisé l'outil segment de droite pour tracer les différents détails contenus dans ce rectangle englobant.

Pendant notre suivi de l'évaluation, nous avons pu mettre en évidence une des causes majeures des échecs enregistrés lors de l'utilisation de l'outil ligne brisée : les évaluateurs n'étant pas habitués au double-clique, ils ne réussissaient pas à arrêter l'utilisation de cet outil (la version d'interface évaluée exigeait d'eux de cliquer deux fois au même endroit). De ce fait, ils dessinent un nombre important de petits segments qui s'intersectent. Et quand ils arrivent à arrêter l'utilisation de l'outil ligne élastique dans leur dessin à main levée, ils oublient d'éliminer du traitement "Carte Planaire" ces petits segments en se servant de la touche clavier 'Delete' qui permet d'annuler le dernier tracé.

Le résultat satisfaisant du dessin libre conforte la remarque des évaluateurs disant que les outils se complètent et explique leurs grandes hésitations avant de faire part de leur préférence pour l'un des quatre outils. Pour certains d'entre eux, nous avons même dû formuler la question différemment en les mettant devant l'éventualité de ne garder qu'un seul des quatre outils (rectangle, ligne droite, ligne brisée ou élastique et main levée) dans l'avenir.

3.2. Résultats de l'expérience 2

Les résultats de cette expérience sont regroupés dans deux tableaux :

- le tableau 2 (cf. annexe D) qui correspond aux conditions dans lesquelles s'est déroulé l'apprentissage de l'outil graphique.
- le tableau 3 (cf. annexe D) qui contient les performances des évaluateurs dans les deux réalisations du test.

Deux de nos évaluateurs (2 chercheurs) n'ont pu participer à cette expérience pour des raisons indépendantes de leur intérêt pour l'outil.

La meilleure performance dans les deux réalisations du test (obtenue par le sujet 3) est certainement due à deux facteurs essentiels :

- le sujet 3 a enregistré la plus longue période d'apprentissage effectif (5 heures et 31 minutes),
- le sujet 3 a suivi un apprentissage bien réparti dans le temps.

Pour interpréter raisonnablement les performances des autres évaluateurs, nous devons prendre en compte d'autres facteurs :

- le temps passé dans la lecture du manuel d'utilisation,
- le temps passé en essais sur machine,
- la durée de la dernière interruption : le nombre de jours entre le dernier contact de l'évaluateur avec l'interface et sa première réalisation du test,
- les performances personnelles de l'évaluateur (expérience ...),
- le degré de perturbation d'un évaluateur causé par notre présence à ses côtés pendant la réalisation du test.

Une interprétation correcte de l'influence de tous ces facteurs nécessiterait la réalisation d'autres expériences avec différents groupes d'évaluateurs pour lesquels on ferait varier un ou plusieurs facteurs. La méthode statistique de l'analyse de la variance permet d'analyser les résultats issus de ce genre d'expériences. Par ailleurs, on aurait pu faire cette analyse par rapport au degré de connaissance de Suntools, en considérant deux groupes d'évaluateurs par exemple. Nous ne l'avons pas fait, car ceux qui ne connaissaient pas Suntools avaient, pour la plupart, déjà travaillé avec le système de fenêtrage X11 qui fonctionne de manière similaire.

De ce fait, il nous a semblé raisonnable de nous baser, pour l'analyse, essentiellement sur les performances moyennes réalisées par l'ensemble des évaluateurs (cf. tableau 3 de l'annexe D).

Nous tenons à signaler qu'à l'issue de la première réalisation du test, nous avons satisfait le désir de certains évaluateurs qui voulaient connaître la manière d'effectuer les tâches pour lesquelles ils avaient échoué. Ceci explique la diminution entre les deux réalisations du test des occurrences globales des tâches où un score nul a été enregistré. A l'exception des tâches 12, 15 et 17, aucune tâche n'a posé problème deux fois à un même sujet. Pour mieux rendre compte de la difficulté que présentaient ces tâches aux évaluateurs, nous les décrivons ci-dessous :

La tâche 12 consistait en la modification d'un faux-plancher en un faux-plafond. Une des difficultés qui a été rencontrée lors de la réalisation de cette tâche est la sélection du faux-plancher antérieurement à sa reclassification ou à sa destruction. Comme la reclassification d'un faux-plancher en un faux-plafond n'avait pas été envisagée lors de la conception de l'interface, les

évaluateurs devaient garder leur sérénité et aboutir à la seule solution possible qui consiste en la destruction du faux-plancher et l'installation d'un faux-plafond à la place. L'essai d'accomplissement de cette tâche a abouti, pour certains évaluateurs, à l'installation d'un faux-plancher et d'un faux-plafond dans la même pièce. Ceci a été considéré comme une erreur, même si l'interface n'effectue aucun contrôle sur ce genre d'action.

La tâche 15 demandait le dessin schématique des connexions liant certains objets aux autres éléments de réseau. Nous avons choisi comme objets des objets icôniques que le système crée automatiquement suite à la spécification de liens entre des objets icôniques. Cette tâche demandait aux évaluateurs un effort et une organisation importantes, et une bonne compréhension du dispositif d'information offert par l'interface.

Ce dispositif permet, en demandant l'information sur un objet donné, de connaître l'identificateur de l'objet ainsi que ceux des objets auxquels il est connecté. La tâche 15 pouvait se résoudre par une série de demandes d'informations sur les objets impliqués dans la connexion dès le début de la recherche ou suite à une demande d'information ultérieure.

Ce shéma des connexions ayant été jugé intéressant par les évaluateurs, certains parmi eux ont suggéré d'intégrer à l'interface une fonctionnalité qui réalise l'équivalent de la tâche 15.

La tâche 17 consistait en une modification éventuelle des dimensions de deux étages antérieurement spécifiés de sorte qu'ils occupent exactement le même espace (boîtes englobantes des étages superposables). Cela nécessitait l'utilisation de la fonctionnalité 'Resize Storey'. Il faut signaler que cette évaluation nous a permis de remarquer le dysfonctionnement de cette fonctionnalité dans certains cas. C'est ce dysfonctionnement qui a été à l'origine des échecs lors de l'exécution de cette tâche par le sujet 1.

3.3. Le questionnaire

Le questionnaire était composé de deux parties : questionnaire 1 et questionnaire 2. Le questionnaire 1 était relatif à la satisfaction visuelle des évaluateurs vis à vis de l'interface. Il devait être rempli au début de l'expérience 1. Il se terminait par une question portant sur le manuel d'utilisation, question qui pouvait n'être remplie qu'à la fin de l'expérience 2. Le questionnaire 2 était lui-même divisé en plusieurs thèmes : les choix propres à l'interface, les choix propres à l'application, l'apprentissage du système et quelques impressions générales.

Le risque souligné dans [SENA 90] a été constaté dans les réponses aux questionnaires. En effet, certaines des réponses des évaluateurs paraissent étranges à première vue. Nous informons le lecteur que les tableaux groupés en annexe D ne permettent pas de relever les informations dont il s'agit dans ce paragraphe en raison de leur organisation de manière à faciliter la lecture des scores moyens. Nous rapportons ici deux exemples opposés du risque en question :

Exemple 1: les scores (réponses) attribués par un évaluateur aux questions du questionnaire 1 n'ont été autres que 8 et 9. Pourtant, cet évaluateur s'est montré raisonnable en ne répondant pas à des questions portant sur des fonctionnalités qu'il n'avait pas essayées.

Exemple 2 : les scores attribués par un évaluateur aux six questions du questionnaire 2 relatives à l'apprentissage de l'interface ont été : cinq fois le score 0 et une fois le score 5. L'évaluateur en question est l'un des deux évaluateurs qui n'attribuent pas un préjugé favorable aux interfaces à manipulation directe.

De ce fait, nous n'allons pas nous appesantir sur les scores individuels mais nous allons nous baser sur les résultats moyens. Cependant, quelques questions ont soulevé des problèmes particuliers que nous commentons dans ce qui suit.

Il y avait une question relative à la facilité d'utilisation des paramètres de la carte planaire. Cette question a eu un score médiocre. Nous pensons que l'ajout d'indications dans le manuel d'utilisation devrait aider à résoudre ce problème. Ces indications seraient plus fructueuses si elles étaient directement liées à la manière d'utiliser les touches clavier 'Delete', 'Back Space' et les flèches de déplacement. Des indications portant sur la façon de coordonner l'utilisation de ces touches de clavier avec celle de la souris seraient aussi d'une grande utilité.

Un autre point noir relatif à l'utilisation de l'interface est lié à l'absence d'information sur ce que le système est en train de faire, et à la difficulté de correction des erreurs. Les mauvais scores engendrés par ce point noir ont été accompagnés de suggestions d'ajout des fonctionnalités :

- 'Undo' permettant de retourner à l'état antérieur,

- 'Save' permettant de sauvegarder un état intermédiaire quelconque du système, et offrant la possibilité d'y retourner à n'importe quel stade de l'interaction.

Deux questions de questionnaire 2 étaient liées à l'aspect direct ou non des protocoles d'accomplissement des tâches. L'une étant intégrée à la partie liée aux choix dans les moyens de dialogue, et l'autre intégrée à la partie évaluant la facilité d'apprentissage. Les scores moyens relatifs à ces deux questions sont différents. Les auteurs de [ROOT 83], référencé dans [SENA 90], recommandent d'utiliser une technique permettant de tester la confiance à accorder aux réponses des évaluateurs. Cette technique consiste à constituer des questionnaires où existent plusieurs formulations de la même question. Nous n'avons pas fait une utilisation systématique de cette technique pour éviter d'allonger davantage notre questionnaire.

Pour finir ce paragraphe, nous donnons le score moyen calculé pour tout le questionnaire : 5.967. Ce score moyen est assez bon si on le situe par rapport au centre de l'échelle des scores, qui vaut 4.5. Mais ce chiffre cache des problèmes spécifiques que les questions ont permis de relever, et également quelques bonnes fonctionnalités.

Dans la section suivante, nous allons décrire les problèmes d'utilisabilité collectés. Et nous renvoyons le lecteur aux annexes pour y trouver les documents qui ont servi à conduire l'évaluation ainsi que les tableaux de résultats commentés.

Ces annexes sont :

Annexe A : le questionnaire,
Annexe B : le test de la fin de l'expérience 2,
Annexe C : le livret d'apprentissage,
Annexe D : les tableaux 1 à 8 de résultats.

3.4. Les problèmes d'utilisabilité et les propositions des évaluateurs

3.4.1. Les problèmes d'utilisabilité

Pendant le déroulement des deux expériences d'évaluation de l'outil graphique développé au sein de l'EMSE, nous avons incité les évaluateurs à nous signaler tout ce qui semblait à leur yeux poser un problème d'utilisabilité. Cette collecte de problèmes réalisée, nous les avons catalogués en nous aidant des critères ergonomiques validés par Bastien [BAST 91].

**** a. Guidage ****

a.1. Prompting

Les évaluateurs se plaignent d'un manque d'assistance dans les tâches qu'ils ont à accomplir :

1- l'interface ne propose pas à l'utilisateur tous les cas possibles de sélection d'objets en cas d'ambiguïté.

2- quand l'utilisateur active une fonctionnalité dans un contexte qui n'est pas le sien, l'interface doit l'assister, par exemple à l'aide d'un message, pour l'activer correctement.

Les évaluateurs reprochent à l'interface de ne pas leur indiquer constamment son état courant :

1- il n'y a pas de retour d'information spécifique à la sélection multiple d'un objet.

2- le pointeur de la souris ne change pas selon que l'on est en mode dessin ou en mode sélection.

a.2. Groupement/Distinction par le format

Les évaluateurs souhaitent voir une différence dans l'aspect graphique entre les différentes classes de bouton :

- les boutons associés à un menu,
- les boutons simples,
- les boutons bloquants (le bouton 'OK' de la fenêtre de message).

a.3. Feed-back immédiat

L'interface n'offre pas les retours d'information que l'on attend suite aux actions entreprises : l'action de placement d'un point à l'aide du pointeur de la souris ne visualise aucun indice au point de l'écran où l'action intervient.

a.4. Clarté

Les évaluateurs ne comprennent pas toujours les représentations icôniques :

1- les icônes des murs perforables et non perforables,

2- l'icône située juste au dessus de l'ascenseur de la zone des éléments de réseau.

Certains indices graphiques ont été difficiles à percevoir par les évaluateurs :

1- la distinction entre une icône sélectionnée et une autre qui ne l'est pas n'est pas facile,

2- la superposition des textures de faux-plafond et de faux-plancher, quand ces deux types d'objets sont installés dans la même pièce, donne une texture qui se rapproche des deux précédentes.

** b. Charge de travail **

b.1. Actions minimales

Les évaluateurs regrettent qu'une optimisation des actions de nature "répétitive" n'ait pas été faite :

1- pour sélectionner plusieurs gaines techniques horizontales, il faut se mettre plusieurs fois en sélection et choisir l'option du menu de sélection "Select shaft",

2- pour installer la même icône de réseau en plusieurs endroits, il faut revenir la sélectionner dans la zone des éléments de réseau à chaque fois.

L'organisation de l'interface fait que certaines fonctionnalités passent par des détours pénalisants :

1- pour désélectionner un objet, on est obligé de désélectionner tous ceux qui ont été sélectionnés après lui (la désélection fonctionne par rapport à l'ordre de sélection selon le mode LIFO),

2- la fenêtre F3D couvrant une bonne partie de la zone de dessin de F2D, les évaluateurs n'apprécient pas le fait de sélectionner les objets depuis la fenêtre F3D pour accéder ensuite au menu de la zone de dessin de F2D permettant d'activer les fonctionnalités de destruction des objets, d'information sur les objets ...

b.2. Charge mentale

Les évaluateurs demandent que la fonctionnalité d'information soit plus développée : actuellement, elle indique seulement l'existence ou non de connexions mais ne fournit rien sur leur nature (l'objet 1 est dans la pièce2).

** c. Actions explicites **

Concernant le cas précis de la fonctionnalité donnant des informations sur un objet sélectionné, les évaluateurs désirent la mise en place d'une action explicite de désélection : une telle modification permettrait à l'interface de demander à l'utilisateur, pour chaque objet sélectionné, s'il veut le libérer ou bien le garder sélectionné. L'utilité de cet arrangement se manifeste dans les nombreux cas où un évaluateur s'informe sur un objet avant d'opérer sur lui une modification importante (connexion, déconnexion, destruction ...).

** d. Adaptabilité **

d.1. Flexibilité

L'interface ne réagit pas convenablement par rapport à des actions équivalentes pour l'utilisateur moyen :

1- la touche clavier 'Return' n'est pas équivalente au bouton bloquant 'OK' de la fenêtre de message,

2- la touche clavier 'Delete' n'est pas équivalente à l'option de menu 'Delete'.

d.2. Prise en compte de l'expérience de l'utilisateur

Les évaluateurs regrettent l'inexistence de raccourcis clavier des commandes et l'inexistence d'un langage de commandes (cela permet à un utilisateur expérimenté d'éviter de parcourir tous les menus de l'interface dans la recherche d'une commande dont il connait l'existence et l'appellation).

Remarque : Cet handicap disparaitrait pour l'interface MMI^2 dans son ensemble puisqu'elle comprend plusieurs modes de dialogue dont un langage de commande.

** e. Gestion des erreurs **

e.1. Protection contre les erreurs

Les évaluateurs trouvent que l'interface n'effectue pas de prévention contre les erreurs :

1- il n'y a pas d'action explicite prévue pour confirmer une demande de destruction,

2- on peut installer autant de faux-planchers et de faux-plafonds qu'on veut dans une même pièce sans que l'interface réagisse.

e.2. Correction des erreurs

L'interface n'offre pas de moyens du type sauvegardes intermédiaires ou fonctionnalité 'Undo' pour permettre aux utilisateurs de corriger certaines erreurs.

Il est difficile de sélectionner une gaine technique horizontale mal installée dans une pièce où une autre gaine est correctement installée. Donc la destruction de la mauvaise installation est impossible.

** f. Homogénéité/Consistence (Consistency) **

Les évaluateurs ont relevé dans l'interface certaines incohérences qui les ont gênées pendant leur apprentissage :

1- l'existence côte à côte de boutons qui noircissent et de boutons qui ne noircissent pas, quand on les sélectionne,

2- l'impossibilité de sélectionner les câbles de connexion entre icônes créés automatiquement, alors qu'ils sont représentés par des segments tout comme les câbles qu'on installe volontairement.

** g. Signification des codes **

L'interface présente des fonctionnalités différentes dont les appellations sont similaires :

1- l'option de menu 'Delete', la touche clavier 'Delete' et le bouton 'Delet',
2- les options du menu 3D 'Install service cable' et 'Put backbone cable'.

** h. Compatibilité **

Certaines actions sur l'interface ne produisent pas les effets qu'attendent les évaluateurs :

1- cliquer plusieurs fois au même endroit dans la zone d'ascenseur ne permet pas de se déplacer plus vite,
2- cliquer sur un bouton sélectionné (un outil de dessin par exemple) ne le désélectionne pas.

L'action permettant d'arrêter l'outil de dessin ligne élastique, n'est pas compatible avec l'aptitude de l'utilisateur moyen (elle demande de cliquer deux fois exactement au même endroit, c'est à dire sans déplacer la souris d'un pixel).

3.4.2. Les propositions d'améliorations dans le manuel

Les propositions d'améliorations dans le manuel sont de trois types :

1- les propositions d'ajout,
2- les propositions de modification dans la terminologie utilisée,
3- les propositions de modification concernant l'organisation adoptée pour le manuel d'utilisation de l'interface graphique.

a. Les ajouts proposés

- la fonctionnalité de désélection, qui n'est pas du tout exposée dans le manuel.

- un sommaire au début du manuel pour permettre une recherche rapide des fonctionnalités.

- un paragraphe concernant la construction des cartes planaires (option de menu 'Construct PM') et ses deux aspects statique et incrémental.

- un paragraphe portant sur l'uniformisation des icônes sur la fenêtre F3D. Cette uniformisation se traduit, pour les icônes de réseau, par la visualisation d'une petite icône carrée. En ce qui concerne les icônes de bâtiment représentant les fonctionnalités verticales (sortie d'escalier, d'ascenseur ou de gaine technique verticale), cette uniformisation se traduit par la visualisation d'une petite icône circulaire.

b. Les modifications concernant la terminologie utilisée dans le manuel

- une préférence des évaluateurs a été notifiée pour parler de 'placement de point' au lieu de 'sélection de point'.

- l'utilisation du mot bouton est exagérée dans le manuel; on suggère d'utiliser : les touches clavier, les touches (droite, milieu, gauche) de la souris et un bouton de l'interface.

- faire coïncider la terminologie du manuel avec celle de l'interface : 'Planar Map' avec 'Construct PM' et 'Service cable' avec 'V.Shaft'.

c. Les modifications concernant l'organisation du manuel

- le contenu de l'introduction utilise des choses pas encore définies (zone de dessin, installation de faux-plancher et de faux-plafond ...). La proposition faite a été de déplacer ce qui était une vue d'ensemble dans notre esprit vers la fin du manuel.

- le paragraphe relatif à la zone de dessin est à mettre au début du manuel.

- le paragraphe relatif au bouton '3D' est prématuré là où il est. Il vaut mieux faire à son niveau un renvoi à un paragraphe ultérieur.

- le manuel doit être adapté au cas du graphique seul (isolé de tous les modes de dialogue de l'interface MMI[2]) : les identificateurs d'objets sous forme de nombres doivent remplacer ceux du manuel qui sont une concaténation de la classe de l'objet et de son numéro de création.

3.4.3. Les propositions de nouvelles fonctionnalités

Les propositions faites sont de deux types :

- celles inspirées par l'interface elle même et ses manques,
- celles inspirées par l'expérience et la connaissance d'autres interfaces.

a. Les propositions inspirées par l'interface elle-même

- l'existence dans les ascenseurs d'une flèche tout en haut pour monter et d'une flèche tout en bas pour descendre, la descente s'effectuant régulièrement ligne par ligne dans la zone des éléments de réseau par exemple.

- le bouton '3D' basculant et permettant aussi bien l'ouverture de la fenêtre '3D' que sa fermeture.

- rendre possible la reclassification d'un câble, antérieurement sélectionné, en choisissant une des options du menu regroupant les différents types de câbles, au lieu de passer par l'option 'Reclassify' du menu de la zone de dessin.

- rendre possible la reclassification des objets 'sortie d'escalier' en 'sortie d'ascenseur' puisqu'ils font partie de la même famille, et faire en sorte que leur installation se fasse de la même manière que les éléments de réseau (c'est à dire pas par paires).

- mettre en place une fonctionnalité qui fait ce que demande la tâche 15 du test (dessin schématique des connexions entre différents objets).

- un problème se pose, dans la pratique, quand l'utilisateur dessine de toutes petites pièces : la sélection des murs constituant une pièce n'est pas équivalente à la sélection de la pièce. Les évaluateurs proposent de faire en sorte que ces deux actions soient équivalentes.

b. Les propositions inspirées par l'expérience d'autres interfaces

- mettre en place une aide en ligne thématique, et fournir une assistance pour les opérations 'Move storey' et 'Resize storey' comme le fait le système de fenêtrage Suntools.

- pour les outils de dessin, ne pas avoir à déplacer la souris avec une de ses touches enfoncée, et exiger par exemple pour le dessin d'un segment avec l'outil segment :

* placement d'une extrémité du segment (pression et relâchement d'une touche de la souris),
* simple déplacement de la souris vers la seconde extrémité,
* placement de la seconde extrémité.

- l'expérience MacDraw de la plupart des évaluateurs les a conduit naturellement à proposer des fonctions du type :

* dupliquer une arête,
* déplacer, allonger, tourner, couper/coller une arête,
* dupliquer des rectangles, dupliquer des étages,

3.4.4. Remarques portant sur certains aspects positifs de l'interface

** a. Clarté **

Les icônes des outils de dessin sont lisibles facilement sans texte associé.

** b. Actions minimales **

Certains évaluateurs préférent l'outil de dessin ligne élastique car, en définitive, cet outil n'oblige pas à cliquer deux fois pour chaque segment dessiné, et est donc moins fatiguant.

** c. Flexibilité **

Les évaluateurs ont trouvé que les quatre outils de dessin supportés par l'interface (segment, ligne brisée, rectangle élastique et dessin à main levée) se complètent; ils ont souligné que leur choix d'utiliser un outil plutôt qu'un autre dépendait du dessin à réaliser.

** d. Compatibilité **

Certains évaluateurs ont compris d'eux mêmes qu'une option de menu grisée est une option déjà validée, et que pour arrêter la ligne brisée il fallait réaliser un double clique.

4. MODIFICATIONS APPORTÉES À L'INTERFACE GRAPHIQUE À LA SUITE DE L' ÉVALUATION

4.1. Modifications concernant le comportement des boutons et des menus de l'interface

4.1.1. Masquage/Affichage des boutons et des menus

Ne sont rendus visibles pour l'utilisateur que les boutons et les menus pouvant être activés dans l'état courant de l'interface graphique :

- tout menu associé à un bouton non activé est masqué. Il en résulte que le menu permettant le choix des contraintes sur les lignes à dessiner n'est visible par l'utilisateur que lorsque les contraintes (horizontale ou verticale) sont significatives. C'est à dire après l'activation de l'un des outils segment et dessin à main levée auxquels nous avons associé ce menu,

- le bouton de la sémantique de gaine technique horizontale n'est visible par l'utilisateur que lorsque l'utilisateur passe en mode de construction incrémentale des cartes planaires en ayant sélectionné la vue bâtiment et réseau,

- les icônes des éléments de réseau et des installations verticales sont considérées comme des **boutons spéciaux** qui ne sont visibles que lorsque l'utilisateur passe en mode de construction incrémentale des cartes planaires,

- certains boutons regroupés en une **barrette** sont visualisés en fonction de l'état courant de l'interface graphique. Cependant, les **boutons de la barrette ne sont pas concurrents**, c'est à dire que l'activation de l'un d'entre eux ne masque pas les autres,

- le bouton de sélection (et le bouton des caractères non encore utilisé) n'est visible par l'utilisateur que lorsque celui-ci passe en mode de construction incrémentale des cartes planaires,

L'effet escompté de ces mesures est de servir de prévention contre les erreurs du type activation d'une fonctionnalité n'ayant pas de sens dans le contexte courant. Elles doivent également rendre plus efficace la recherche des fonctionnalités par l'utilisateur grâce à une optimisation du nombre de boîtes de dialogue visualisées à la fois. En pratique, cela constitue une double protection contre les erreurs. En effet, l'utilisateur est déjà averti, dans le manuel d'utilisation, de toutes les conditions de lancement des fonctionnalités. En cas d'oubli, cette façon d'organiser les moyens de dialogue sert de second rempart.

4.1.2. Activation/Désactivation d'un bouton - Ouverture/Fermeture d'un menu

Dans ce qui suit, nous reprenons les notations des diverses actions de l'utilisateur (cf. section 2.2. de la partie C du chapitre III).

Tout bouton de l'interface graphique peut être activé/désactivé du moment qu'il est visible par l'utilisateur.

L'activation/désactivation d'un bouton de l'interface se fait par une action adg en son emplacement. L'activation d'un bouton masque les boutons qui lui sont concurrents, sa désactivation provoque leur réapparition pour l'utilisateur (outils de dessin par exemple).

L'activation d'un bouton provoque l'ouverture du menu associé (si menu il y a) et le noircissement du bouton en question.

La désactivation d'un bouton provoque la fermeture du menu associé (si menu il y a) et rend normal son aspect graphique.

Tout menu est associé à un bouton (l'inverse n'est pas vrai).

La fermeture d'un menu sans choix d'option se fait également par une action add en son intérieur; cela provoque aussi la désactivation du bouton auquel il est associé.

4.1.3. Les boutons de la barrette

Les boutons de la barrette ont, chacun, un menu associé. Cette barrette est placée en haut à droite de F2D : fenêtre principale de l'interface graphique. Il n'y a donc plus besoin de faire des allers/retours entre la fenêtre principale et la fenêtre F3D dans la recherche d'une option de menu.

A un état donné de l'interface correspond au plus un menu ouvert de la barrette.

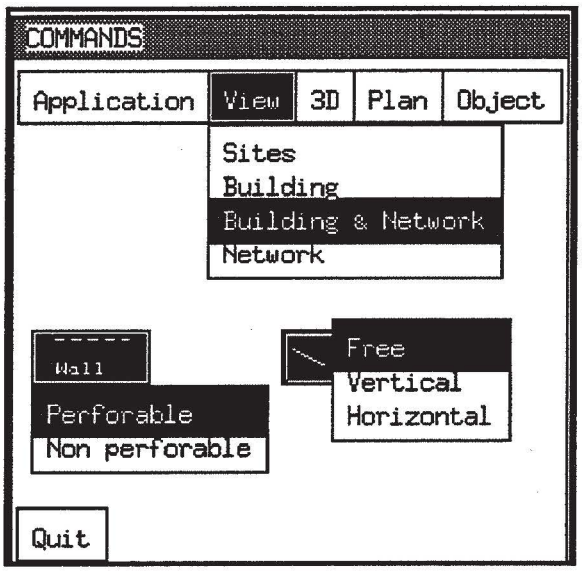


Figure 79 : la zone des commandes au commencement d'un dessin

Le menu permettant de choisir l'une des vues possibles (bâtiment d'un immeuble, réseau d'un immeuble, bâtiment et réseau d'un immeuble, vue de l'ensemble des immeubles : site) fait partie des menus de la barrette (cf. figure 79). Le menu des vues peut être ouvert, avec le choix courant noirci, alors que l'un des boutons (sélection, dessin ou caractère) est activé. Ceci est le contraire de ce qui se passe pour les autres menus de la barrette afin de fournir à l'utilisateur une information continue sur la vue courante.

L'état de la barrette dépend du choix courant de la vue : site ou pas (cf. figure 80).

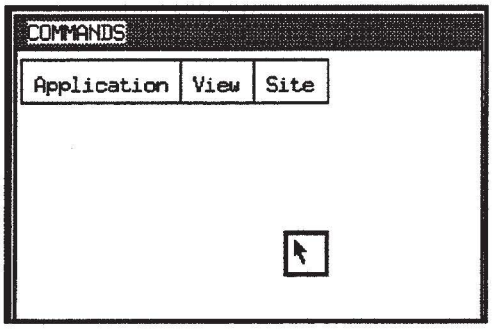


Figure 80 : état de la barrette pendant la vue de l'ensemble des immeubles (sites)

4.1.4. Le bouton de sélection/désélection des objets et son menu

Les besoins actuels de l'application ont fait qu'il n'y a pas réellement besoin de faire des sélections multiples. Cela nous a permis d'autoriser la désélection directe d'un objet par sa resélection. Dans la version antérieure de l'interface, un utilisateur ne pouvait désélectionner que l'objet placé en tête de la pile des objets sélectionnés.

Une sélection multiple, si besoin est, peut être résolue par la mise en place d'un paramètre nouveau intitulé nombre_de_sélections. La gestion de la zone 'PARAMETERS' pour les quelques états du système rencontrés prouve que cela est possible (cf. figure 82).

L'activation du bouton de sélection ouvre le menu associé. Ce menu dépend de la vue courante et permet de désigner par sa classe l'objet concerné par la sélection parmi tous les objets se trouvant dans la même zone de l'écran que lui (cf. figure 81). L'intérêt de cette organisation est de lever l'ambiguïté dans la fonctionnalité de sélection.

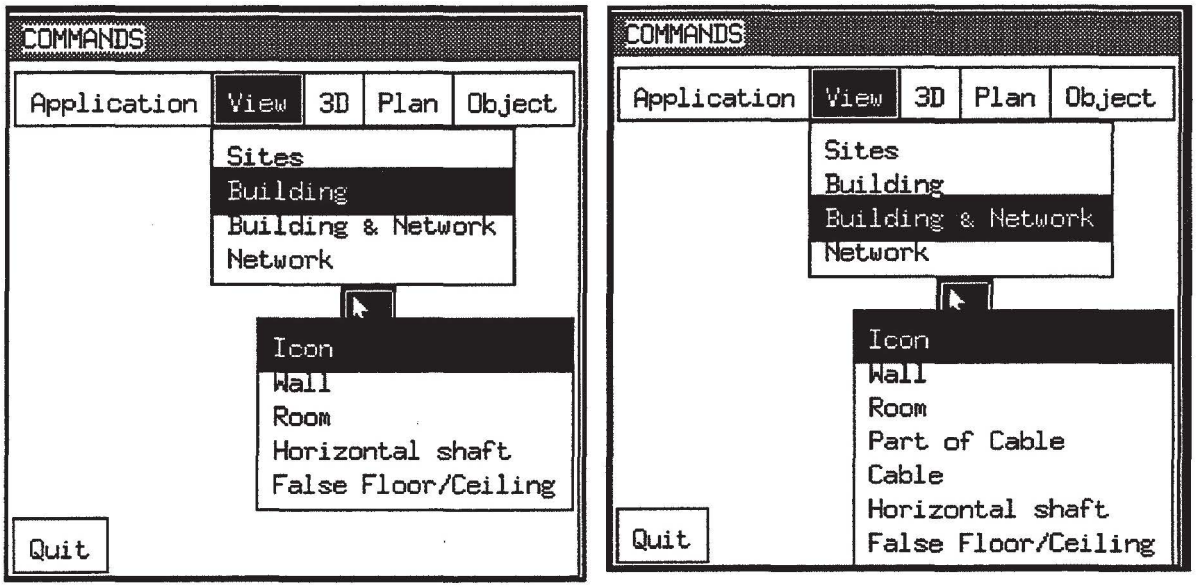


Figure 81 : le bouton de sélection et la dépendance de son menu par rapport à la vue courante

4.1.5. La gestion des boutons spéciaux

Rappelons que les boutons spéciaux sont les icônes des éléments de réseau (affichées dans la zone 'NETWORK COMPONENTS') et les icônes des installations verticales (affichées dans la zone 'BUILDING ICONS').

L'activation d'un bouton spécial désactive le dernier bouton activé, que ce soit dans la zone 'NETWORK COMPONENTS' ou dans celle intitulée 'BUILDING ICONS'.

L'activation d'un bouton spécial se fait, comme pour les autres boutons, par une action adg. Cela provoque son noircissement.

Aucun objet icônique associé à une des icônes des boutons spéciaux ne peut être installé sans que :

- le bouton spécial associé soit activé,
- le commencement de l'opération dite de port soit spécifié.

Le commencement de l'opération de port se spécifie par une action adm au dessus d'un bouton spécial activé dans l'état courant.

La fin de l'opération de port se spécifie à l'aide d'une action add en n'importe quelle zone de l'interface graphique, et uniquement comme cela.

Ceci permet par exemple d'installer le même objet icônique en plusieurs endroits sans revenir activer le bouton spécial associé à chaque fois. C'est une optimisation des tâches répétitives.

4.2. Modifications liées à l'utilisation de la zone des paramètres

Les paramètres visibles dans cette zone dépendent de l'état courant de l'interface graphique (cf. figure 82).

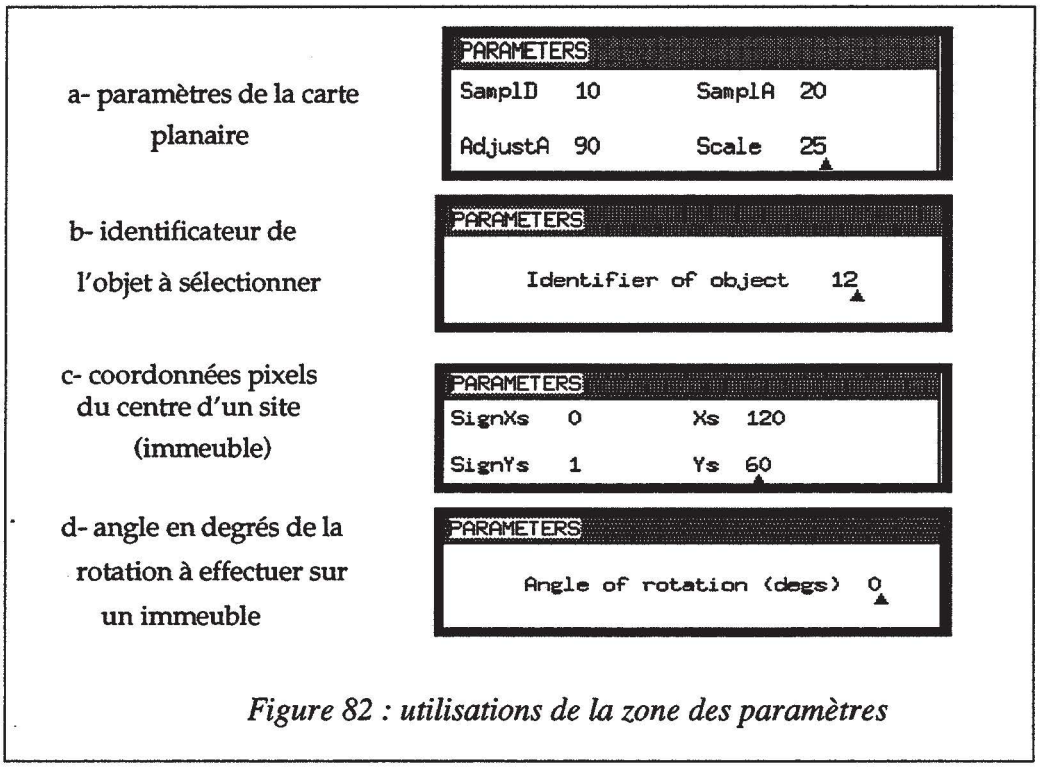


Figure 82 : utilisations de la zone des paramètres

Ces paramètres peuvent bien sûr être modifiés par l'utilisateur. Il doit d'abord effectuer une action adg à l'emplacement où se trouve le champ à modifier. Ensuite, l'effacement de la valeur courante se fait par utilisation des touches clavier 'Delete' et 'Back Space' : 'Delete' permet de détruire le premier chiffre du champ situé à droite de la matérialisation du curseur; 'Back Space' permet de détruire le premier chiffre du champ situé à gauche de la matérialisation du curseur

4.3. Modifications liées à l'utilisation des outils de dessin

Chacun des outils de dessin est associé à un bouton (rectangle, segment, main levée et ligne brisée). Les outils de dessin à main levée et de segment peuvent être utilisés dès que le bouton associé est activé. L'utilisateur peut alors choisir de contraindre les lignes dessinées à être verticales ou horizontales en se servant du menu associé aux boutons (cf. figure 79).

Les outils de dessin rectangle élastique et ligne brisée peuvent être utilisés dès que le bouton associé est activé. Ces boutons n'ont pas de menu associé.

Nous avons décrit dans la section 2.3.2. de la partie C du chapitre III comment nous avons codé les actions d'un utilisateur se servant de l'un de ces outils pour saisir un dessin.

La nouvelle manière de gérer les outils de dessin a plusieurs avantages:

- éviter de garder une touche de la souris enfoncée, ce qui est de nature à fatiguer les utilisateurs dès qu'une certaine limite est dépassée,

- épargner aux utilisateurs de cliquer deux fois au même endroit dans le but d'arrêter le dessin d'une ligne brisée,

- diminuer le nombre de petits segments dessinés involontairement est une conséquence indirecte des deux avantages précédents. Ceci doit en principe diminuer le nombre d'échecs enregistrés après le lancement de la construction statique d'une carte planaire.

4.4. Modifications liées aux moyens d'information de l'utilisateur

Nous avons effectué des modifications dans le but de résoudre des problèmes précis soulevés pendant l'évaluation.

Par exemple, dès qu'un utilisateur sélectionne un objet et demande les informations qui lui sont attachées, la fenêtre de message lui est proposée avec toutes les informations disponibles et permet de libérer l'objet en question ou de le garder sélectionné (cf. figure 83).

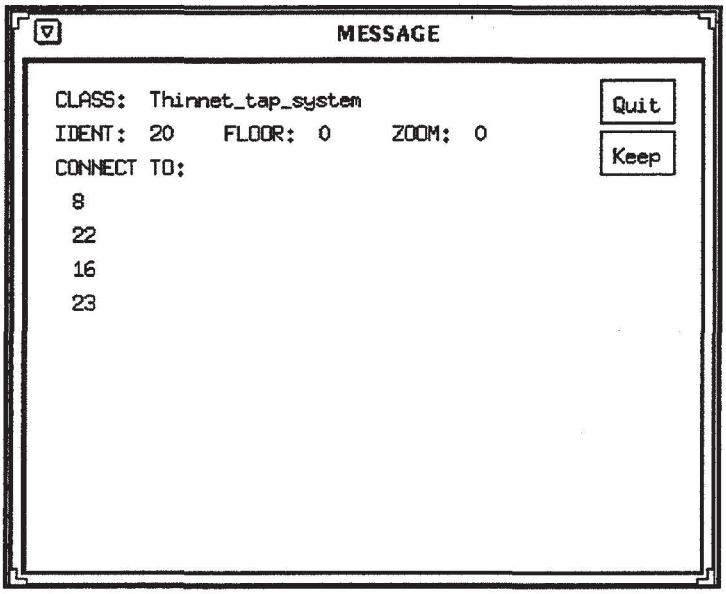


Figure 83 : information sur un objet sélectionné ou désigné par son identificateur

Si les informations sur un objet font intervenir des identificateurs d'autres objets obj_k, l'utilisateur peut se renseigner sur les objets obj_k sans qu'ils soient visibles, et sans qu'il sache exactement les situer dans l'écran d'affichage. Cela peut être réalisé en activant la fonctionnalité d'information sans sélection préalable d'objets. Une fenêtre de message le guide pour remplir le paramètre nécessaire (identificateur de l'objet) pour désigner l'objet (cf. figure 82(b)).

4.5. Modifications en vue d'une bonne prévention contre les erreurs

Nous avons renforcé l'utilisation de la fenêtre de message, et nous avons créé un message en chacune des situations où une information ou une alerte de l'utilisateur est nécessaire. Cela a été fait par exemple pour la fonctionnalité de destruction où une confirmation de l'utilisateur a été rendue nécessaire préalablement à son exécution (cf. figure 84).

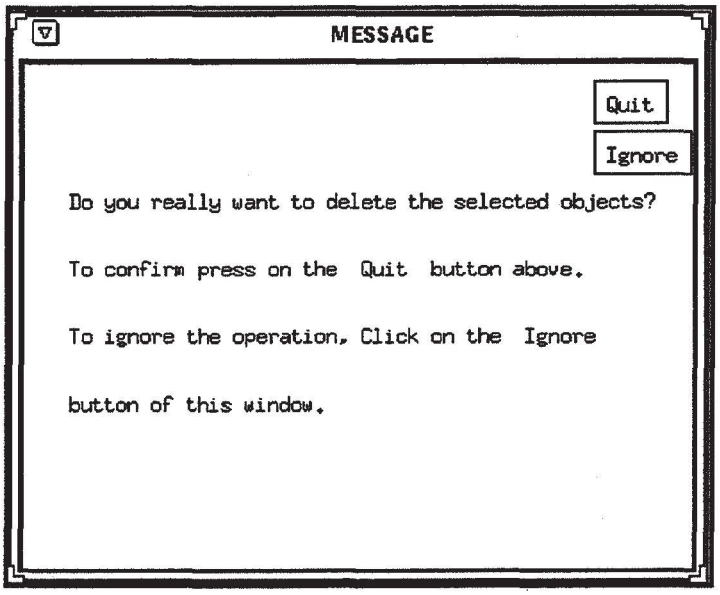


Figure 84 : message demandant à l'utilisateur une confirmation de destruction d'objets

D'autre part, quand une saisie de données passe par des champs de saisie de la fenêtre de message, nous avons établi un dialogue où une confirmation finale est faite explicitement par l'utilisateur. Lorsque l'utilisateur choisit une option à travers la fenêtre de message, ce choix est noirci comme retour à son action (cf. figure 85).

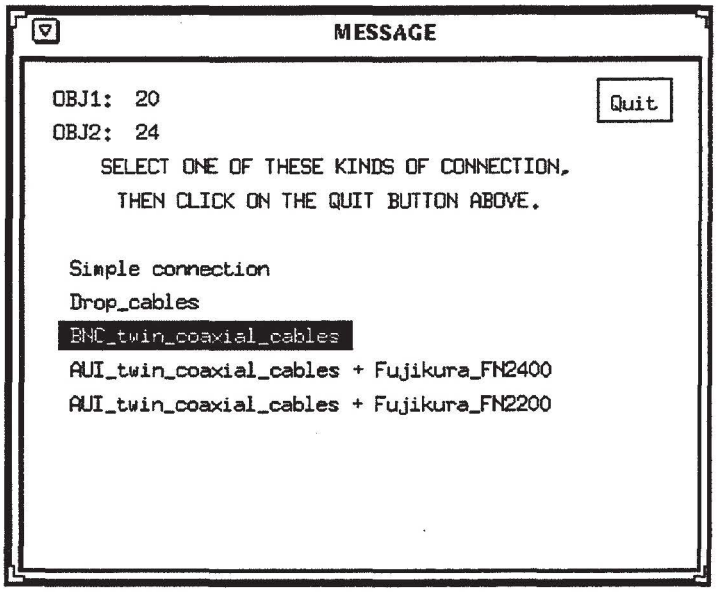


Figure 85 : message invitant l'utilisateur à choisir un type de lien et à confirmer après par appui sur le bouton 'Quit'

5. CONCLUSION

Une étude bibliographique sur les techniques d'évaluation des interfaces homme-machine a été présentée dans ce chapitre. C'est à la lumière de cette étude que nous avons choisi la technique d'évaluation du premier prototype de l'interface graphique développée dans notre laboratoire. La technique d'évaluation, qui se compose de deux étapes, a été exposée et son choix justifié. La première étape de l'évaluation nous a révélé que l'utilisation des outils de dessin peut être maîtrisée en peu de temps et de façon satisfaisante. La deuxième étape, approfondissant l'apprentissage de l'ensemble des outils de manipulation directe, a également donné de bons résultats : les scores obtenus par les évaluateurs dans leurs soumissions au test sont très satisfaisants et la durée moyenne de l'apprentissage est acceptable. L'étude bibliographique nous a appris, nous mêmes, à bien observer les évaluateurs pendant leurs interactions avec l'interface et à les encourager à nous fournir leurs suggestions et remarques. Nous avons pu ainsi collecter un certain nombre de problèmes d'utilisabilité et les classer à l'aide de critères ergonomiques. L'application présentée dans le chapitre III est sans doute plus agréable en utilisation par rapport au premier prototype. Si elle l'est, elle le doit en priorité à la participation active des évaluateurs que nous remercions.

CONCLUSION GÉNÉRALE

Pendant nos années de thèse, nous nous sommes intéressé à deux thèmes différents qui s'inscrivaient de façon naturelle dans le cadre d'un projet européen Esprit de recherche et de développement : la modélisation basée sur les cartes planaires d'une part, les interfaces homme-machine et leur évaluation d'autre part. Autrement dit, cette thèse expose des travaux de nature théorique et d'autres de nature pratique :

- la partie théorique est une modélisation $2D^{1/2}$ hiérarchique basée sur les cartes planaires,

- la partie pratique se compose de l'élaboration de modules du premier prototype d'une interface graphique, de la réalisation d'une évaluation ergonomique de cette interface et enfin de l'apport d'améliorations à cette interface.

L'apport théorique de la thèse est double :

- une structure de données $2D^{1/2}$ qui repose sur l'empilement de cartes planaires représentant des données d'un même niveau plan ou de niveaux plans strictement parallèles,

- une structure de données hiérarchique qui permet non seulement de décrire des données à différents niveaux de détail mais aussi d'assurer la continuité géométrique de données appartenant à des niveaux différents.

Ces deux aspects de notre modèle sont simples de conception et reposent tous les deux sur une structure de données modélisant les liens sémantiques.

La superposition des cartes planaires est utilisée de telle sorte que plusieurs sémantiques puissent être modélisées au sein d'une même application. La structure 'liens' permet de préserver le sens global de données appartenant au même niveau plan et qui, au regard de leur sémantique, sont modélisées par des cartes planaires indépendantes. Nous empilons de façon presque similaire des cartes planaires modélisant des données appartenant à des plans strictement parallèles. La structure 'liens' permet, là encore, d'établir des relations du type 'en dessous de', 'au dessus de' entre les objets de ces plans.

Nous retirons de l'empilement et de la superposition des cartes planaires des avantages du point de vue de l'affichage : affichage des cartes planaires d'une sémantique particulière, simulation 3D par une perspective 'cavalière' présentant des plans horizontaux strictement parallèles dans des zones de l'écran qui ne se chevauchent pas ...

Dans notre conception de la hiérarchie et des niveaux de détail, nous admettons la représentation multiple des objets. La structure de données hiérarchique proposée dans la thèse a trois avantages :

- regrouper des données selon leur sémantique, en différentes cartes planaires, et offrir comme moyen de regroupement leur appartenance à une zone particulière : face (s) de la carte planaire du niveau de détail juste inférieur à celui représenté par les dites données,

- ne pas encombrer l'écran de détails inutiles pour un contexte donné et permettre à l'utilisateur d'organiser lui même ses données sous forme de hiérarchie,

- utiliser la structure 'liens' pour maintenir liées, de façon systématique, les différentes représentations d'un même objet physique.

L'étude bibliographique portant sur l'évaluation des interfaces homme-machine n'est pas la première dans son genre. Les travaux portant sur l'évaluation sont, en effet, abondants. Cette étude est une compilation de publications portant sur les aspects théoriques ou pratiques de l'évaluation. Notre travail sur ce thème s'est inspiré des différentes publications référencées. Suite à cette étude, nous avons fixé notre méthode d'évaluation qui adapte les techniques rapportées à nos objectifs et à l'interface à évaluer. L'avantage de notre méthode est qu'elle permet d'évaluer différents aspects d'une même interface par des moyens naturels.

RÉFÉRENCES BIBLIOGRAPHIQUES

[ANSA 85] S. Ansaldi, L. De Floriani, B. Falcidieno, "Geometric modeling of solid objects by using a face adjacency graph representation", SIGGRAPH'85, ACM Computer Graphics, Vol. 19, No 3, July 1985, 131-139.

[ANSI 88] "American National Standard for Human Factors Engineering of Visual Display Terminal Workstations", Human Factors Society, Santa Monica, CA, USA, ANSI/HFS, Standard No. 100-1988, 1988.

[ARQU 92] D. Arquès, N. Janey, "Modélisation de cartes planaires pour la synthèse d'images de reliefs montagneux", MICAD'92, Paris, February 1992, 247-262.

[ANTI 90] J. F. Antin , T. A. Dingus, M. C. Hulse, W. W. Wierwille, "An evaluation of the effectiveness and efficiency of an automobile moving map navigational display", International Journal of Man-Machine Studies, 33, 1990, , 581-594.

[AYAL 85] D. Ayala, P. Brunet, R. Juan, I. Navazo, "Object representation by means of NonMinimal division Quadtrees and Octrees", ACM Transactions on graphics, 4(1), January 1985, 41-59.

[BAST 91] C. Bastien, "Validation de critères ergonomiques pour l'évaluation d'interfaces utilisateurs", Rapport de Recherche INRIA No 1427 , Mai 1991.

[BAUD 89] P. Baudelaire, M. Gangnet, "Planar maps : an interaction paradigm for graphic design", CHI'89 Proceedings, May 1989, 313-318.

[BAUM 72] B. G. Baumgart, "Winged-edge polyhedron representation", Stanford Artificial Intelligence Report No CS-320, 1972.

[BAUM 75] B. G. Baumgart, "A polyhedron representation for computer vision", AFIPS Conference, 1975, 589-596.

[BEIG 89] M. Beigbeder, B. Péroche, "Un système de synthèse d'images 3D : ILLUMINES", Actes des journées UNIX de Grenoble, Octobre 1989, 263-276.

[BEIG 91] M. Beigbeder, G. Jahami, "Managing levels of detail with textured polygons", Compugraphics'91, Sesimbra, Portugal, September 1991, 479-489.

[BELA 82] A. Belaid, G. Masini, "Segmentation de tracés manuscrits sur tablette graphique en vue de leur reconnaissance", TSI, Vol. 1, No 2, 1982, 155-168.

[BENA 92a] H. Ben Amara, "Un outil graphique pour une interface multi-modes", Thèse Ecole Nationale Supérieure des Mines de Saint-Etienne, Saint-Etienne, Mars 1992.

[BENA 92b] H. Ben Amara, A. Nahed, B. Péroche, "Un outil graphique pour une interface multi-modes", INFORSID'92, Clermont-Ferrand, Mai 1992, 347-366.

[BENO 93] M. O. Benouamer, "Opérations booléennes sur les polyédres représentés par leurs frontières et imprécisions numériques", Thèse Ecole Nationale Supérieure des Mines de Saint-Etienne, Saint-Etienne, Juillet 1993.

[BENO 93a] M. O. Benouamer, P. Jaillon, D. Michelucci, J. M. Moreau, "A lazy exact arithmetic library", 11th. IEEE Symposium on Computer Arithmetic, Windsor, Ontario, Canada, 30 June - 2 July 1993.

[BENT 79] J. L. Bentley, T. A. Ottmann, "Algorithms for reporting counting geometric intersections", IEEE Transaction on computer, Vol. 28, No 9, 1979, 643-647.

[BERG 58] C. Berge, "Théorie des graphes et ses applications", Dunod, Paris, 1958.

[BERG 83] C. Berge, "Graphes", Editions Gauthier-Villars, 1983, p.16.

[BERT 79] M. Berthod, P. Jancenne, "Le pré-traitement des tracés manuscrits sur tablette graphique", AFCET meeting, Reconnaissance des formes et intelligence artificielle, Toulouse, Septembre 1979, 195-209.

[BERT 92] Y. Bertrand, J. F. Dufour, J. Françon, P. Lienhardt, "Modélisation volumique à base topologique", MICAD'92, Paris, February 1992, 59-74.

[BIM 89] "Common Meaning Representation", BIM/13, deliverable d2, Esprit Project No 2474 MMI^2, November 1989.

[BINO 88] J. L. Binot, B. Demoen, K. H. Hanne, L. Solomon, Y. Vassiliou, W. von Hahn, T. Wachtel, "LOKI : a Logic Oriented Approach to data and Knowledge bases supporting natural language Interaction", in Esprit'88 Conference Proceedings, North-Holland, 1988.

[BINO 90] J. L. Binot, P. Falzon, R. Perez, B. Péroche, N. Sheehy, J. Rouault, M. D. Wilson, "Architecture of a multimodal dialogue interface for knowledge based systems", ESPRIT P2474 MMI^2, in Esprit'90 Conferernce Proceedings, Kluwer Academic Publishers, Dordrecht, 1990, 412-434.

[BRAQ 88] J. P. Braquelaire, P. Guitton, "A model for image structuration", Computer Graphics International'88, Genève, May 1988, 426-435.

[BRAQ 90] J. P. Braquelaire, P. Guitton, "Structuration de contours discrets par la modélisation $2^{1/2}$D", Actes des journées AFCET-GROPLAN Algorithmique graphique et géométrique, BIGRE 67, Janvier 1990, 29-34.

[BRAQ 91] J. P. Braquelaire, P. Guitton, "2D$^{1/2}$ scene update by insertion of contour", Computer & Graphics, Vol. 15, No 1, 1991, 41-48.

[BROW 88] Brown, "Human-Computer interface design guidelines", Norwood, NJ: Albex, 1988.

[BUSA 65] R. G. Busacker, T. L. Saaty, "Finite graphs and networks : an introduction with applications", International Series in pure and applied mathematics, McGraw-Hill Book Company, 1965.

[CARD 80] S. K. Card, T. P. Moran, "The keystroke-level model for user performance time with interactive systems", Commun. ACM 23, 1980, 396-410.

[CARD 83] S. K. Card, T. P. Moran, A. Newell, "The psychology of Human-Computer Interaction". Lawrence Erlbaum Associates, Hillsdale, N.J. , 1983.

[CHAP 91] H. Chappel, M. Wilson, "The graphical presentation of structured representation", deliverable d38, Esprit Project No 2474 MMI^2, June 1991.

[CHEN 92] M. Chen, P. Townsend, C. Y. Wang, "A development environment for constructing graph-based editing tools", Eurographics'92, Vol. 11, No 3, 1992, 345-355.

[CHIN 88] J. P. Chin, V. A. Diehl, K. L. Norman, "Development of an Instrument Measuring User Satisfaction of the Human-Computer Interface", CHI'88, 1988, 213-218.

[COQU 84] S. Coquillart, "Représentation de paysages et tracé de rayon", Thèse Ecole Nationale Supérieure des Mines de Saint-Etienne, Saint-Etienne, 1984.

[CORI 84] R. Cori, "Cartes, hypercartes et leurs groupes d'automorphismes", Séminaire Lotharingien, Université de Bordeaux I, UER de Mathématiques et d'Informatique, 1984.

[DAMI 91] S. Dami, "The french syntactic analyser", Esprit Project No 2474 MMI^2, October 1991.

[DAMI 93] S. Dami, "The french generation system", deliverable d10, Esprit Project No 2474 MMI^2, July 1993.

[DARS 90] F. Darses, F. Balfroid, C. Jouve, "A Multi-Mode Interface for Man-Machine Interaction with knowlegde based system; knowledge acquisition", deliverable d43, Esprit Project No 2474 MMI^2, May 1990.

[DARS 91] F. Darses, P. Falzon, "Evaluating the MMI^2 system" , Technical Report, Esprit Project No 2474 MMI^2, April 1991.

[DeFL 93] L. De Floriani, D. Mirra, E. Puppo, "Extracting contour lines from a hierarchical surface model", EUROGRAPHICS'93, Vol. 12, No 3, 1993, 249-260.

[DIN 87] DIN 66234 Part 8 : Principles of dialogue design. 1987.

[DOME 91] J. P. Domenger, "Gestion du positionnement des objets d'une scène 2D$^{1/2}$", Journées graphiques GROS PLAN 91, Décembre 1991, 81-89.

[DUFO 88] J. F. Dufour, "Construction of interactive programs in computer graphics", Computer Graphics Forum, Vol. 7, No 3, September 1988, 161-176.

[EDMO 60] J. Edmonds, "A combinatorial representation for Polyhedral surfaces", Notices of Amer. Math. Soc., 7, 1960.

[ELTE 91] H. Elter, P. Lienhardt, "Extension de la notion de carte pour la représentation de la topologie d'objets géométriques complexes", Journées graphiques GROS PLAN 91, Décembre 1991, 149-156.

[FINI 89] T. Finin, "GUMS, a General User Modeling Shell", A. Kobsa and W. Whalser Eds., User Models in Dialog Systems, Springer Verlag, Berlin, 1989.

[GANG 89] M. Gangnet, J. C. Hervé, T. Pudet, J. M. Van Thong, "Incremental computation of planar maps", Computer Graphics, Vol. 23, No 3, July 1989, 345-354.

[GIBB 92] S. C. Gibbons, "Towards an expert system based menu interface evaluation tool", Rapport de Recherche INRIA No 1581, January 1992.

[GONG 90] R. Gong, J. Elkerton, "Designing Minimal Documentation Using a GOMS Model : A Usability Evaluation of an Engineering approach", CHI'90 Proceedings, April 1990, 99-106.

[GREE 86] D. H. Greene, F. Yao, "Finite-resolution computational geometry", Proc. of the 27th. annual Symposium of the Foundations of Computer Science, 1986, 143-152.

[GREE 93] N. Greene, M. Kass, G. Miller, "Hierarchical Z-Buffer Visibiliy", Proc. SIGGRAPH'93, ACM Computer Graphics, August 1993, 231-238.

[GUIB 85] L. Guibas, J. Stolfi, "Primitives for the manipulation of general subdivisions and the computation of Voronoï diagrams", Transactions on graphics, 4(2), ACM, 1985, 74-123.

[HART 90] H. R. Hartson, A. C. Siochi, D. Hix, "The UAN : A User-Oriented Representation for Direct Manipulation Interface Designs", ACM Transactions on Information systems, Vol. 8, No. 3, July 1990, 181-203.

[HAYE 88] N. A. Hayes, D. E. Broadbent, "Two models of learning for interactive tasks", Cognition, 22, 1988, 249-275.

[HELA 88] M. Helander, "Handbook of Human-Computer Interaction", Amsterdam : Elsevier, 1988.

[HEND 89] J. J. Hendrickson, "Performance, preference, and visual scan patterns on a menu-based system : implications for interface design", CHI'89 Proceedings, May 1989, 217-222.

[HERO 76] C. Herot, "Graphical input through machine recognition of sketches", Computer Graphics, Vol. 10, No 2, 1976, 97-102.

[HOPP 93] H. Hoppe, T. DeRose, T. Duchamp, "Mesh Optimization", Proc. SIGGRAPH'93, ACM Computer Graphics, August 1993, 19-26.

[HOWA 87] S. Howard, D. M. Murray, "A taxonomy of evaluation techniques for HCI", Human-Computer Interaction - INTERACT'87, 1987, 453-459.

[HOWE 91] M. Howes, N. Ghali, N. Sheehy, K. Mardia, M. Wilson, "Gesture Mode", deliverable d32, Esprit Project No 2474 MMI^2, April 1991.

[JAHA 91] G. Jahami, B. Péroche, "Un modeleur flexible et évolutif orienté objet pour la synthèse d'images", Actes de MICAD'91, 282-297.

[JACQ 70] A. Jacques, "Constellations et graphes topologiques", Combinatorial Theory and Applications, Budapest,1970, 657-673.

[JEFF 91] R. Jeffries, J. R. Miller, C. Wharton, K. M. Uyeda, "User interface evaluation in the real world : a comparison of four techniques", CHI'91 proceedings, Conference on Human Factors in Computer systems, New York : Association for Computing Machinery, 1991, 119-124.

[JOUV 92] C. Jouve, "Représentation des connaissances pour les problèmes de conception. Application à un système à base de connaissances pour la conception de réseaux informatiques : NEST", Thèse Ecole Nationale Supérieure des Mines de Saint-Etienne, Saint-Etienne, Septembre 1992.

[KIER 85] D. E. Kieras, P. G. Polson, "An approach to the formal analysis of user complexity", International Journal of Man-Machine Studies, 22, 1985, 365-394.

[KRIS 88] R. Krishnamurti, "Representational semantics for graphical discourse", ESPRIT/ACORD, Task 3.6 deliverable, EdCAAD report, January 1988.

[KUIJ 91] E. Kuijpers, G. Lalliche-Boidin, J. Rouault, "French system : from a syntactic structure to a CMR expression", Esprit Project No 2474 MMI^2, May 1991.

[LAID 86] D. H. Laidlaw, W. B. Trumbore, J. F. Hugues, "Constructive Solid Geometry for polyhedral objects", Proc. SIGGRAPH'86, ACM Computer Graphics, Vol. 20, No 4, August 1986, 161-180.

[LIEN 88] P. Lienhardt, "Subdivisions of surfaces and generalized maps", Rapport interne No 88-11, Université Louis Pasteur de Strasbourg, Octobre 1988.

[LUND 85] M. A. Lund, "Evaluating the User Interface: The Candid Camera Approach", CHI'85 Proceedings, April 1985, 107-113.

[MACK 89] W. E. Mackay, T. W. Malone, K. Crowston, R. Rao, D. Rosenblitt, S. K. Card, "How do experienced Information Lens users use rules", CHI'89 Proceedings, May 1989, 211-216.

[MANT 82] M. Mantyla and R. Sulonen, "GWB : A solid modeler with Euler operators", IEEE CG & A, Vol. 2, No 7, Septembre 1982, 18-31.

[MICH 84] D. Michelucci, M. Gangnet, "Un outil graphique interactif", MICAD 84, Hermès, Février-Mars 1984, 95-110.

[MICH 84a] D. Michelucci, M. Gangnet, "Communication homme-machine et dessin des projets", Rapport final relatif à l'exécution d'un projet, Ecole Nationale Supérieure des Mines de Saint-Etienne, Septembre 1984.

[MICH 87] D. Michelucci, "Les représentations par les frontières", Thèse Ecole Nationale Supérieure des Mines de Saint-Etienne, Saint-Etienne, Novembre 1987.

[MOIS 84] J. C. Moissinac, "Aides informatiques à la réalisation de dessins animés", Thèse Ecole Nationale Supérieure des Mines de Saint-Etienne, Saint-Etienne, 1984.

[MOLI 90] R. Molish, J. Nielsen, "Improving a Human-Computer Dialogue", Communications of the ACM, 33/3, March 1990, 338-348.

[MORA 81] T. P. Moran, "The Command Language Grammar : A representation of the user interface of interactive computer systems", International Journal of Man-Machine Studies, 15, 1981, 3-50.

[MORE 90] J. M. Moreau, "Hiérarchisation et facétisation de la représentation par segments d'un graphe planaire dans le cadre d'une arithmétique mixte", Thèse Ecole Nationale Supérieure des Mines de Saint-Etienne, Saint-Etienne, Octobre 1990.

[MULL 78] D. E. Muller, F. P. Preparata, "Finding the intersection of two convex polyhedra", Theoritical Computer Science 7(2), 1978, 217-236.

[NIEL 90] J. Nielsen, R. Molish, "Heuristic Evaluation of User Interfaces", CHI'90 Proceedings, April 1990, 249-256.

[NOVA 87a] F. Novara, N. Bertaggia, N. Allamano, "Usability evaluation and feedback to designers - an experimental study", In Human-Computer Interaction INTERACT'87, H. J. Bulinger , B. Shackel (Ed.), Elsevier Science Publishers B. V. , North-Holland, 1987, 337-340.

[NOVA 87b] F. Novara, N. Bertaggia, A. Dillon, J. Bonner, "The evaluation of products using the usability methodology with proposals for product development", working paper A5.3a, ESPRIT Project 385- HUFIT/04-OLI-11/87, 1987.

[PAOL 89] A. Paoluzzi, M. Ramella, A. Santarelli, "Boolean algebra over linear polyhedra", CAD Vol. 21, No 8, October 1989, 474-484.

[PAOL 93] A. Paoluzzi, F. Bernardini, C. Cattani, V. Ferrucci, "Dimension-independent modeling with simplicial complexes", ACM Trans. on graphics, 12, January 1993, 56-102.

[PLAI 92] C. Plaisant, B. Shneiderman, "Scheduling home control devices : design issues and usability evaluation of four touchscreen interfaces", International journal of Man-Machine Studies, 36, 1992, 375-393.

[POLS 92] P. G. Polson, C. Lewis, J. Rieman, C. Wharton, "Cognitive walkthroughs : a method for theory-based evaluation of user interfaces", International journal of Man-Machine Studies, 36, 1992, 741-773.

[PREP 88] F. P. Preparata, M. I. Shamos, "Computational Geometry : An Introduction", Springer-Veralg, Berlin, 1988.

[REIS 83] P. Reisner, "Formal grammars as a tool for analysing ease of use : some fundamental concepts", in J. C. Thomas, M. Shneider (Eds.), Human factors in computing systems. Norwood, NJ : Ablex, 1983.

[RIVL 90] C. Rivlin, R. Lewis, R. D. Cooper, "Guidelines for screen design", Oxford, England : Blackwell Scientific, 1990.

[ROMM 87] E. Rommel, "Modélisation de terrains polyplissés et polyfaillés", Mémoire de Diplôme d'ingénieur, Département de Géologie, Ecole Nationale Supérieure des Mines de Saint-Etienne, Juin 1987.

[ROY 69] B. Roy, "Algèbre moderne et théorie des graphes", tome 1, Donod, Paris, 1969.

[SALV 87] G. Salvendy, "Handbook of human factors", Canada : Wiley & Sons, 1987.

[SCAP 90] D. L. Scapin, "Organizing human factors knowledge for the evaluation and design of interfaces", International Journal of Human-Computer Interaction, 2(3), 1990, 203-229.

[SCAP 87] D. L. Scapin, "Guide ergonomique de conception des interfaces homme-machine", Rapport de Recherche INRIA No 77, 1987.

[SENA 90] B. Senach, "Evaluation ergonomique des interfaces homme-machine : une revue de la littérature", Rapport de Recherche INRIA No 1180, Mars 1990.

[SHAR 87] B. Sharatt, "The incorporation of early interface evaluation into Command Language Grammar specification", in People and Computers III, D.Diaper and R.Winder (Eds.), Cambridge university press : Cambridge, 1987, 11-28.

[SHNE 87] B. Shneiderman, "Designing the user interface : strategies for effective human computer interaction", Readings, MA : Addison-Wesley, 1987.

[SMIT 86] S. L. Smith, J. N. Mosier, "Design guidelines for the user interface software", rapport ESD-TR-86-278, Bedford, Massachusetts : Mitre, 1986.

[SVEN 91] G. B. Svendsen, "The influence of interface style on problem solving", International journal of Man-Machine Studies, 35, 1991, 379-397.

[WALK 91] N. Walker, J. B. Smelcer, E. Nilsen, "Optimizing speed and accuracy of menu selection : a comparison of walking and pull-down menus", International journal of Man-Machine Studies, 35, 1991, 871-890.

[WALK 88] N. Walker, J. R. Olson, "Designing keybindings to be easy to learn and resistant to forgetting even when the set of commands is large", CHI'88, 1988, 201-206.

[WILL 84] B. Williges, R. C. Williges, "Dialogue design considerations for interactive computer systems", in F.A. Muckler (Ed.) The human factors review. The human Factors Society, Santa Monica, 1984.

[WILS 91] M. D. Wilson, "The First MMI^2 Demonstrator : a Multi-modal Interface for Man Machine Interaction with Knowledge Based systems". RAL-91-093, United Kingdom. Deliverable d7, Esprit Project No 2474 MMI^2, December 1991.

[YOUN 89] R. N. Young, T. R. G. Green, T. Simon, "Programmable User Models for predictive evaluation of interface designs", CHI'89 Prodeedings, May 1989, 15-19.

ANNEXES

Annexe A
Le questionnaire

Le questionnaire

Il est constitué de deux parties : le questionnaire 1 (Q1) et le questionnaire 2 (Q2). Q1 est relatif à la satisfaction visuelle et se termine par une question portant sur le manuel d'utilisation de l'interface graphique. Q2 porte sur d'autres thèmes qui sont: les choix relatifs aux moyens de dialogue, les aspects liés à l'application, les aspects liés à l'apprentissage et enfin les impressions générales. Il a été demandé aux évaluateurs d'inscrire au verso du questionnaire leurs remarques, et d'encercler pour chaque question un chiffre entre 0 et 9 correspondant au degré de leur satisfaction.

Le questionnaire 1

IMPRESSIONS VIS A VIS DE L'INTERFACE

Lisibilité de l'écran (à remplir juste avant l'expérience 1)

les caractères sur l'écran sont									
difficiles à lire							faciles à lire		
0	1	2	3	4	5	6	7	8	9
les icônes évoquent naturellement ce qu'elles représentent									
pas du tout							parfaitement		
0	1	2	3	4	5	6	7	8	9
le découpage de la fenêtre de l'outil graphique est									
génant							naturel		
0	1	2	3	4	5	6	7	8	9
la charge visuelle à laquelle est soumis l'utilisateur est									
excessive							normale		
0	1	2	3	4	5	6	7	8	9
les zones d'affichage se distinguent des autres zones de l'écran									
difficilement							facilement		
0	1	2	3	4	5	6	7	8	9
les zones de saisie se distinguent des autres zones de l'écran									
pas du tout							parfaitement		
0	1	2	3	4	5	6	7	8	9

la représentation graphique des ascenseurs est inadéquate adéquate 0 1 2 3 4 5 6 7 8 9
le changement de curseur suivant les zones de l'écran facilite la tâche pas du tout beaucoup 0 1 2 3 4 5 6 7 8 9

Question sur le manuel (juste après l'expérience 1)

le manuel d'utilisation de l'outil graphique prête à confusion est clair 0 1 2 3 4 5 6 7 8 9

Le questionnaire 2

Choix effectués pour les fonctionnalités

l'emploi de la langue anglaise pour nommer les fonctionnalités (commandes ..) est génant naturel 0 1 2 3 4 5 6 7 8 9
le découpage des fonctionnalités en menu, commandes, boutons .. est incohérent cohérent 0 1 2 3 4 5 6 7 8 9
l'existence de menus sur certains boutons est génante naturelle 0 1 2 3 4 5 6 7 8 9
les fonctionnalités supportées par l'interface ont des appelations qui les représentent mal parfaitement 0 1 2 3 4 5 6 7 8 9
la démarche à suivre pour lancer une fonctionnalité est sans détours rarement souvent 0 1 2 3 4 5 6 7 8 9

la syntaxe des opérations est incohérente cohérente 0 1 2 3 4 5 6 7 8 9
la mise en oeuvre des paramètres (SamplA, SamplD, ..) de la notion de carte planaire est difficile facile 0 1 2 3 4 5 6 7 8 9
l'utilisation des boutons de la souris est incohérente cohérente 0 1 2 3 4 5 6 7 8 9
les affichages successifs dans la fenêtre principale se déroulent de façon génante naturelle 0 1 2 3 4 5 6 7 8 9
l'utilisation de la vidéo inverse facilite la tâche pas du tout beaucoup 0 1 2 3 4 5 6 7 8 9
l'utilisation d'une fenêtre spéciale dite de message est inutile utile 0 1 2 3 4 5 6 7 8 9
l'utilisation de la fenêtre de message pour guider l'entrée de données prête à confusion est claire 0 1 2 3 4 5 6 7 8 9
la transformation choisie pour simuler l'aspect tri-dimensionnel sur les arêtes est inappropriée appropriée 0 1 2 3 4 5 6 7 8 9
l'image tri-dimensionnelle des icônes est inappropriée appropriée 0 1 2 3 4 5 6 7 8 9
l'utilisation d'une fenêtre spéciale pour l'aspect tri-dimensionnel est génante naturelle 0 1 2 3 4 5 6 7 8 9

la sélection des objets sur la fenêtre 3 D est difficile facile 0 1 2 3 4 5 6 7 8 9
le fait que les possibilités d'entrée de données soient restreintes en 3 D par rapport à 2 D est génant naturel 0 1 2 3 4 5 6 7 8 9
le fait d'avoir deux menus : un en 2 D et un autre en 3 D est génant naturel 0 1 2 3 4 5 6 7 8 9
les affichages successifs dans la fenêtre 3 D se déroulent de façon génante naturelle 0 1 2 3 4 5 6 7 8 9
la limitation du nombre d'étages visibles à la fois sur la fenêtre 3 D à trois est génante naturelle 0 1 2 3 4 5 6 7 8 9
l'accès aux différents étages d'un immeuble se fait de façon génante naturelle 0 1 2 3 4 5 6 7 8 9
toute l'information présentée sur la fenêtre 3 D se lit difficilement facilement 0 1 2 3 4 5 6 7 8 9
le partage des fonctionnalités entre les fenêtres 2 D et 3 D est incohérent cohérent 0 1 2 3 4 5 6 7 8 9

PARTIE DU QUESTIONNAIRE RELATIVE A L'APPLICATION

l'emploi de la langue anglaise pour nommer les objets de l'application est

	génant							naturel	
0	1	2	3	4	5	6	7	8	9

la terminologie utilisée dans le système est

	incohérente							cohérente	
0	1	2	3	4	5	6	7	8	9

le système vous tient informé de ce qu'il est en train de faire

	jamais							toujours	
0	1	2	3	4	5	6	7	8	9

le système fait appel à des notions indépendantes de l'application

	toujours							jamais	
0	1	2	3	4	5	6	7	8	9

les conséquences des erreurs commises pendant l'utilisation sont immédiatement apparentes

	jamais							toujours	
0	1	2	3	4	5	6	7	8	9

la correction des erreurs est

	difficile							facile	
0	1	2	3	4	5	6	7	8	9

APPRENTISSAGE DU SYSTEME

apprendre à utiliser le système est

	difficile							facile	
0	1	2	3	4	5	6	7	8	9

l'exploration de toutes les particularités du système par essai et erreur est

	difficile							facile	
0	1	2	3	4	5	6	7	8	9

la mémorisation des noms des commandes est

	difficile							facile	
0	1	2	3	4	5	6	7	8	9

la mémorisation du protocole de lancement des commandes est difficile facile 0 1 2 3 4 5 6 7 8 9
le système prend en considération les utilisateurs inexpérimentés rarement souvent 0 1 2 3 4 5 6 7 8 9
le système prend en considération les utilisateurs expérimentés rarement souvent 0 1 2 3 4 5 6 7 8 9

IMPRESSIONS GENERALES SUR LE SYSTEME

la charge mentale à laquelle est soumis l'utilisateur est excessive normale 0 1 2 3 4 5 6 7 8 9
les réactions du système sont trop lentes assez rapides 0 1 2 3 4 5 6 7 8 9
les réactions du système sont prévisibles rarement souvent 0 1 2 3 4 5 6 7 8 9
l'utilisation du système est difficile facile 0 1 2 3 4 5 6 7 8 9
le système est déprimant satisfaisant 0 1 2 3 4 5 6 7 8 9
le système est ennuyeux stimulant 0 1 2 3 4 5 6 7 8 9
le système est rigide flexible 0 1 2 3 4 5 6 7 8 9

Annexe B
Le test de la fin de l'expérience 2

Le test de la fin de l'expérience 2

Après la période d'apprentissage, chaque évaluateur a été soumis à un test constitué de 20 tâches. Avant d'inviter l'évaluateur à se mettre devant la machine où l'interface graphique était installée, nous avions dessiné à l'aide de l'interface en cours d'évaluation, un réseau informatique incomplet dans un bâtiment ne possédant qu'un seul niveau. La figure 1 est une copie d'écran correspondant à l'état de l'interface graphique au début du test. Les tâches à réaliser ont été données aux évaluateurs l'une après l'autre, et non sous une forme imprimée comme cette annexe.

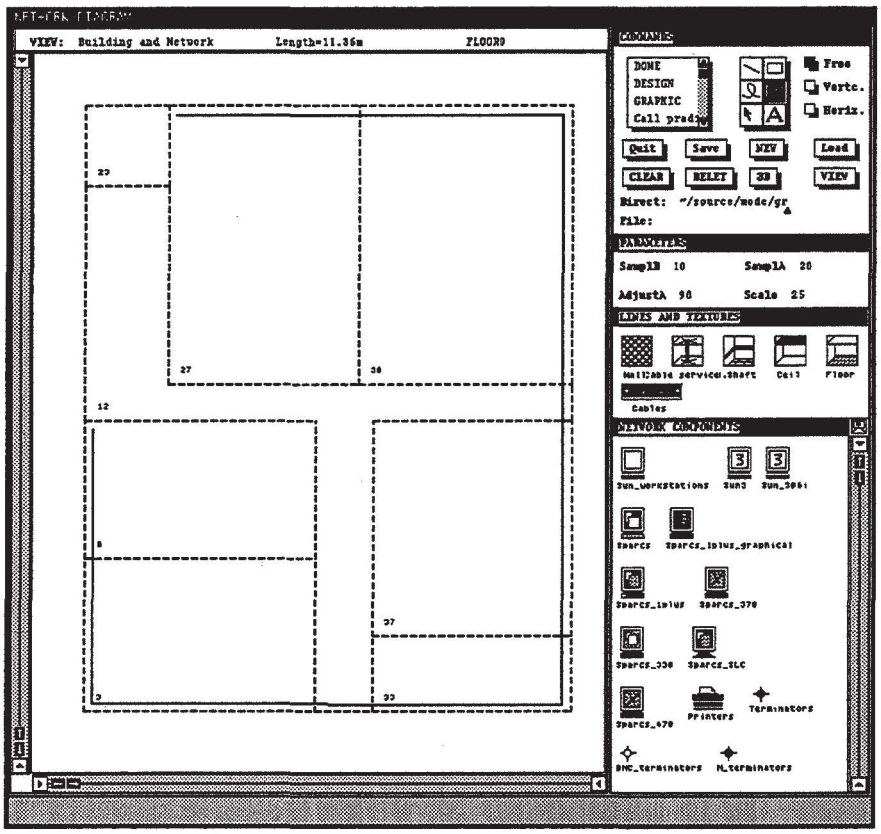


figure 1: état de l'interface au début du test

Tâche 1 : Quel type de câble a été utilisé dans ce réseau ?

Tâche 2 : Faire en sorte que les câbles utilisés soient du type fin.

Tâche 3 : Quel est le type du mur séparant les pièces No 27 et 30 ?
Faire en sorte que ce mur soit de l'autre type possible.

Tâche 4 : Installer deux éléments de réseau appelés "Terminator" à chaque extrémité du câble.
Installer une gaine technique horizontale le long du câble.

Tâche 5 : Installer un élément de réseau appelé "Sparc_SLC" dans la pièce 37.

Tâche 6 : Installer un élément de réseau appelé "Disk_327Mb" dans la pièce 37.

Tâche 7 : Installer un élément de réseau appelé "Thinnet_tap_system" sur le câble dans la pièce 37.

Tâche 8 : Spécifier des connexions simples entre les éléments :
"Sparc_SLC" et "Disk_327Mb"
"Sparc_SLC" et "Thinnet_tap_system".

Tâche 9 : Changer la pièce d'identificateur 12 en un couloir.

Tâche 10 : Installer un faux-plancher dans le couloir.

Tâche 11 : Sélectionner la gaine technique horizontale.

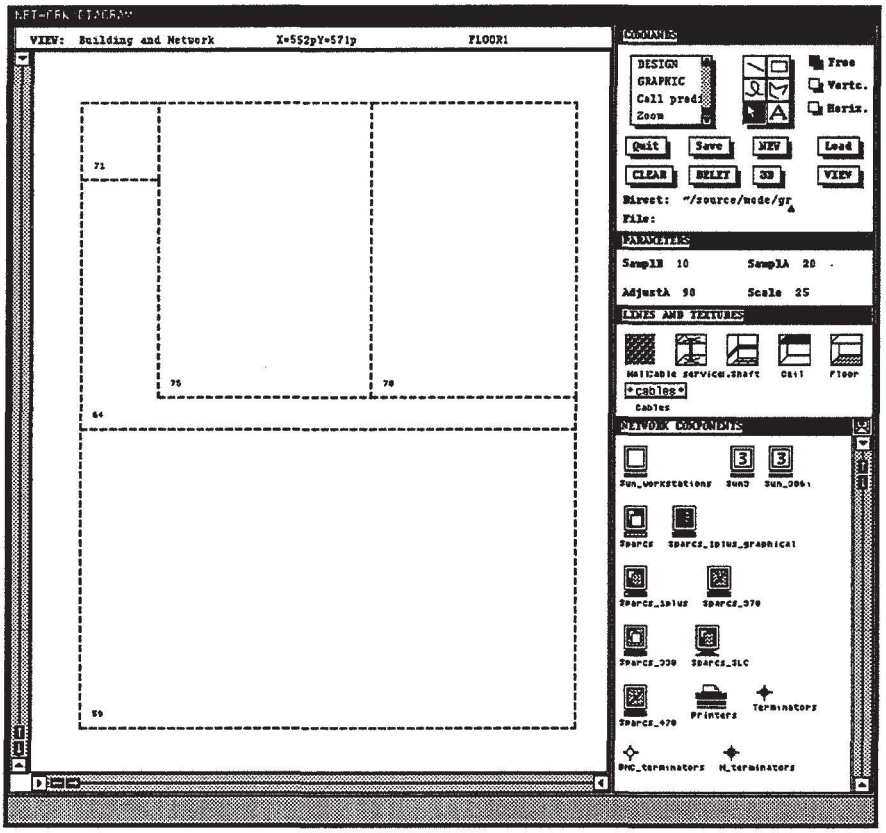


figure 2: la spécification du nouvel étage

Tâche 12 : Remplacer le faux-plancher par un faux-plafond.

Tâche 13 : Y a -t- il des éléments non affichés ?

Tâche 14 : Fournir le type et l'identificateur de chacun de ces éléments.

Tâche 15 : Dessiner sur une feuille de papier le shéma des connexions qui lient :

1- ces éléments,
2- ces éléments aux autres éléments installés.

Tâche 16 : Créer un autre étage correspondant aux spécifications données dans la figure 2.

Tâche 17 : Changer les dimensions des étages de sorte qu'ils correspondent au dessin de la figure 3.

Tâche 18 : Installer des sorties d'escaliers liant les deux étages à travers les pièces identifiées 23 et 71.

Tâche 19 : Aligner les sorties d'escaliers.

Tâche 20 : Passer un câble backbone par ces sorties d'escaliers.

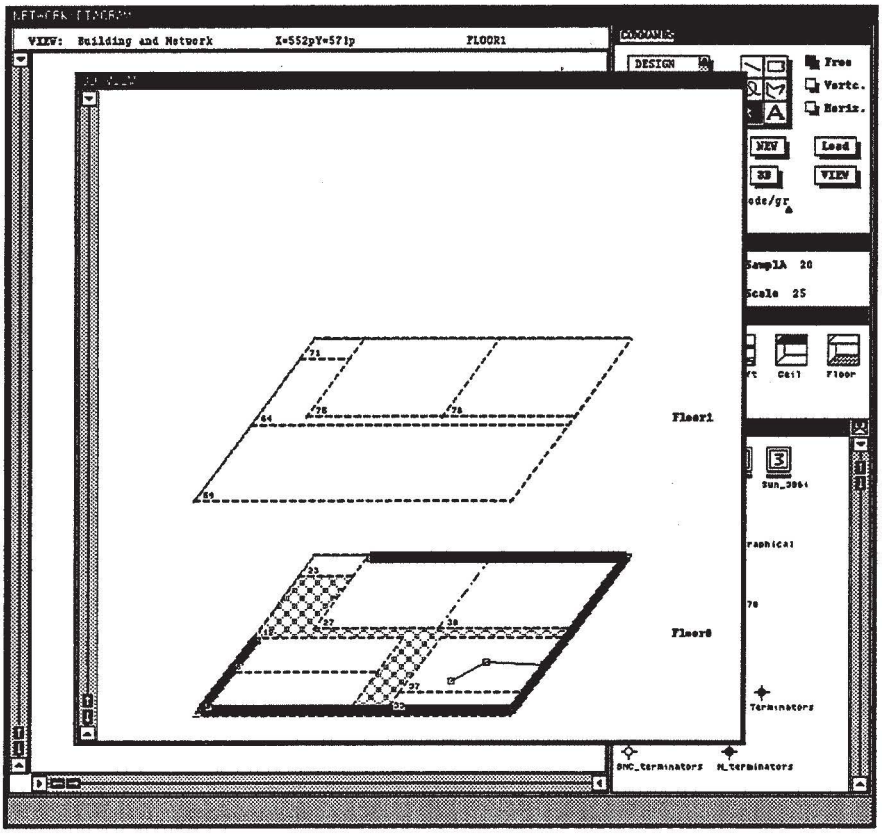


figure 3 : les étages après modification de leurs dimensions

Annexe C
Le livret d'apprentissage

. Remplir les colonnes des jours en fonction du type d'apprentissage que vous avez eu parmi les trois proposés (lignes).

. Ne pas oublier d'inscrire la date du jour dès la première fois de la journée où vous entreprenez l'apprentissage de l'interface.

. Le jour j1 coincide pour vous avec la date du début de la première étape d'évaluation.

Les données sont à remplir en minutes. Exemple pour un jour j : 12

65

Merci pour votre participation.

j1=	jour j1	j2	j3	j4	j5	j6	j7	j8	j9	j10	j11	j12	j13	j14	j15	j16	j17	j18	j19	j20	j21
Lecture du manuel																					
Essais sur machine																					
Autres : Contemplation de l'écran ou des copies d'écran ...																					

Livret d'apprentissage de l'interface graphique

nom:
prénom:
date de début :

Annexe D
Les tableaux 1 à 8 de résultats

Tableau 1

	connaissait Suntools	résultat de l'exécution du dessin avec un seul des outils (minutes)				résultat de l'exécution du dessin librement avec les outils (occurrences)				résultat dessin libre (minutes)	préférence verbale exprimée pour l'outil
		ligne brisée	rectangle	main levée	segment	ligne brisée	rectangle	main levée	segment		
sujet 1	oui	Echec^{3}	3^{2}	4^{4}	2^{1}	2	1	0	9	Echec	segment
sujet 2	non	Echec^{3}	5^{1}	Echec^{4}	2^{2}	0	9	0	4	3	segment
sujet 3	oui	Echec^{3}	2^{1}	3^{4}	Echec^{2}	0	1	0	11	3	segment
sujet 4	oui	5^{3}	2^{1}	5^{4} inc.	4^{2}	0	1	0	10	2 inc.	ligne brisée
sujet 5	non	5^{3}	5^{1}	4^{4}	2^{2}	2	1	0	7	2	segment
sujet 6	non	5^{3} inc.	2^{1}	4^{4} inc.	5^{2}	0	4	0	6	2	rectangle
sujet 7	non	3^{3}	2^{1}	2^{4}	3^{2}	0	0	1	13	2	segment
sujet 8	non	4^{3}	2^{4}	Echec^{1} con.	Echec^{2} con.	1	1	0	9	3	ligne brisée
sujet 9	oui	Echec^{3}	3^{1}	Echec^{4}	4^{2} con.	0	1	0	11 con.	2	segment
sujet 10	oui	3^{3}	5^{1}	Echec^{4}	2^{2}	0	5	0	5	2	main levée

Le tableau 1 résume les résultats de l'expérience 1. Le code 'inc.' indique que l'évaluateur a utilisé la facilité de l'interface qui permet de traiter incrémentalement les tracés qu'il réalise. Tandis que le code 'con.' indique que l'évaluateur s'est servi de la fonctionnalité qui contraint les lignes tracées à être horizontales ou verticales pour ne pas prendre des directions quelconques.

Les sujets ont réalisé le dessin de l'expérience 1 avec un seul des quatre outils (rectangle, segment de droite, ligne brisée et dessin à main levée) dans un ordre arbitraire dicté par la personne qui effectuait le suivi de l'évaluation. Cet ordre varie d'un sujet à l'autre, et sur ce tableau on peut lire cet ordre en exposant dans les quatre colonnes associées aux réalisations du dessin avec un seul des outils. Quant à la base dans ces colonnes, elle indique la durée en minutes de la réalisation du dessin avec un seul des outils. Un échec signifie que le sujet n'a pas pu finir la réalisation correcte du dessin de l'expérience 1 en moins de cinq minutes.

Tableau 2

	nombre de jours entre la 1ère expérience et le 1er test (durée totale de l'apprentissage)	cumul des minutes de lecture du manuel	cumul des minutes d'apprentissage sur machine	cumul des minutes d'apprentissage sur des écrans	Interruptions (IT) pendant l'apprentissage en jours	plus petite IT pendant l'apprentissage	plus grande IT pendant l'apprentissage	moyenne des IT pendant l'apprentissage	nombre de jours entre le 1er test et le 2ème test
sujet 1	9	13	107	15	1,5,0	0	5	2	8
sujet 2	29	40	206	0	0,26,0	0	26	8.67	7
sujet 3	22	118	188	25	0,2,2,3,3,1,0,3	0	3	1.75	8
sujet 4	23	72	36	0	21,0	0	21	10.5	7
sujet 5	22	60	107	10	22,0	0	22	11	61
sujet 6	28	84	110	0	3,20,2	2	20	8.33	50
sujet 7	35	31	88	0	2,31	2	31	16.5	38
sujet 8	28	58	113	6	8,17,0	0	17	8.33	42
moyennes	24.5	59.5	119.4	7				8.38	27.6

Ce tableau regroupe les conditions de déroulement de l'apprentissage de l'interface. Nous remarquons qu'en général les sujets ont consacré deux tiers de leur temps effectif d'apprentissage à des essais sur machine, l'autre tiers étant dédié à la lecture du manuel. Seul le sujet 4 échappe à cette règle générale en passant deux tiers de son temps d'apprentissage en lecture du manuel d'utilisation, et l'autre tiers en essais sur machine.

L'autre moyen d'apprentissage possible (regarder seulement l'interface ou des copies d'écrans en cours d'utilisation) a été utilisé, mais pas de façon significative.

La distribution des temps d'apprentissage à travers le temps est donnée par les interruptions (IT). Une interruption 0 correpsond à un temps consacré à l'apprentissage de l'interface par un évaluateur le jour et le jour qui suit.

Tableau 3

	durée en minutes du 1er test	score du 1er test noté sur 20 points (1 point par tâche)	score nul dans le 1er test pour les tâches:	durée en minutes du 2ème test	score du 2ème test noté sur 20 points (1 point par tâche)	score nul dans le 2ème test pour les tâches:
sujet 1	35	16	13, 14, 15, 17	30	16	17, 18, 19, 20
sujet 2	50	16	12, 13, 14, 15	20	19	15
sujet 3	30	19	12	20	20	néant
sujet 4	68	16	15, 18, 19, 20	30	18	15, 16
sujet 5	42	15	11, 13, 14, 15, 20	42	19	1
sujet 6	60	19	12	43	19	1
sujet 7	45	19	15	50	17	18, 19, 20
sujet 8	70	18	1, 12	58	19	12
scores globaux	50	17.25	$1^1, 11^1, 12^4, 13^3, 14^3, 15^5, 17^1, 18^1, 19^1, 20^3$	36.625	18.375	$1^2, 12^1, 15^2, 16^1, 17^1, 18^2, 19^2, 20^2$

Le tableau 3 résume les performances des évaluateurs pendant l'expérience 2. Si l'on s'intéresse à la performance moyenne on peut dire : après un temps d'apprentissage égal à trois heures et six minutes, réparti sur 24.5 jours, le test (cf. annexe B) a été réalisé pour la première fois en un temps moyen de 50 minutes et le score moyen y a été de 17.25/20. L'interruption moyenne entre les deux réalisations du test a été de 27.6 jours. Après cette interruption, la seconde réalisation du test a duré en moyenne 36 minutes et 37 secondes et le score moyen de 18.375/20 y a été enregistré.

Dans la dernière ligne de ce tableau, les exposants dans les 3ème et 6ème colonnes correspondent au nombre global d'occurrences où un score nul a été enregistré pour une même tâche.

Tableau 4

note / N° Quest.	0	1	2	3	4	5	6	7	8	9	moy.
1	0	0	0	0	2	0	3	3	1	1	6.4
2	1	0	0	0	1	1	2	3	0	2	6
3	0	0	0	0	0	1	1	0	4	4	7.9
4	0	0	0	0	0	0	2	2	3	3	7.7
5	1	0	0	0	0	1	1	1	3	3	6.9
6	0	0	0	0	1	1	3	1	2	2	6.8
7	1	0	0	0	0	1	0	1	3	4	7.2
8	1	0	1	1	0	3	3	0	0	1	4.7
9	0	0	0	0	1	3	2	2	2	0	6.1
occurrences	4	0	1	1	5	11	17	13	18	20	

Ce tableau regroupe les scores attribués par les évaluateurs aux questions du questionnaire 1. On peut y lire pour une ligne donnée le nombre d'occurrences respectives du score 0, du score 1 ... du score 9. Ce tableau montre également que toute l'échelle de notation entre 0 et 9 a été utilisée.

Les scores moyens sont satisfaisants mis à part la question relative au changement du curseur selon la zone de l'interface graphique où se trouve le pointeur de la souris. Les évaluateurs n'ont pas trouvé cela utile, mais ils ont suggéré de changer l'aspect du curseur suivant l'état de l'interface : tracé ou bien sélection.

Tableau 5

N°Quest. \ note	0	1	2	3	4	5	6	7	8	9	moy.
1	1	0	0	0	1	2	2	1	1	0	5.175
2	0	0	0	2	0	0	0	4	1	1	6.375
3	0	0	2	0	1	0	2	1	1	1	5.5
4	0	0	1	1	0	1	2	2	1	0	5.5
5	0	0	0	2	2	3	0	1	0	0	4.5
6	0	0	1	0	2	0	1	3	0	1	5.75
7	0	0	2	0	1	1	1	1	0	0	4.33[3]
8	1	0	1	0	1	1	0	1	2	1	5.375
9	0	0	0	0	1	0	1	3	1	2	7.125
10	1	0	0	0	0	0	1	1	2	3	7
11	0	0	0	0	0	0	1	1	2	4	8.125
12	0	0	0	0	0	0	1	2	4	1	7.625

N°Quest. \ note	0	1	2	3	4	5	6	7	8	9	moy.
13	0	0	0	0	0	0	0	3	4	1	7.75
14	0	0	0	1	0	1	3	1	1	0	5.857[1]
15	1	0	0	0	0	0	0	2	3	2	7
16	1	0	0	0	1	3	0	0	3	0	5.375
17	0	1	0	0	0	2	2	1	1	1	5.875
18	1	0	1	0	0	1	0	2	2	1	5.75
19	0	0	0	1	0	0	0	4	3	0	6.875
20	0	0	0	1	1	0	0	2	2	1	6.571[1]
21	0	0	0	0	0	2	0	1	4	1	7.25
22	0	0	0	1	0	1	0	5	0	1	6.5
23	0	0	0	1	0	0	1	3	3	0	6.75

Le tableau 5 récapitule les scores associés aux questions du questionnaire 2 traitant des choix dans les moyens de dialogue de l'interface. Nous désirons apporter une précision sur des informations de ce tableau : dans la colonne des moyennes, l'exposant indique le nombre d'évaluateurs qui ont refusé d'attribuer un score à l'aspect de l'interface que traite la question. Cela s'est produit pour la septième question du questionnaire 2 où deux refus ont été enregistrés. Dans ces cas, la moyenne inscrite en base dans la colonne des moyennes a été calculée en ne tenant compte que des sujets qui ont attribué un score à la question.

Tableau 6

note / N°Quest.	0	1	2	3	4	5	6	7	8	9	moy.
1	1	1	0	1	1	2	0	1	1	0	4.125
2	0	0	0	0	1	1	2	2	1	0	6.143[1]
3	0	0	1	1	1	3	1	0	0	0	4.286[1]
4	0	0	1	1	0	1	2	2	0	0	5.143[1]
5	0	0	1	2	1	0	1	1	0	2	5375
6	1	0	1	0	1	3	1	1	0	0	425

Le tableau 6 résume les scores des questions du questionnaire 2 sur les aspects de l'interface liés à l'application. Les scores moyens relatifs à ces questions ne sont pas très satisfaisants.

Tableau 7

N°Quest. \ note	0	1	2	3	4	5	6	7	8	9	moy.
1	1	0	0	0	1	3	1	1	1	0	5
2	1	0	0	0	1	3	2	1	0	0	4.75
3	0	0	0	1	2	1	0	1	1	2	6.125
4	1	0	0	0	1	0	1	1	2	1	6[1]
5	1	0	0	1	2	2	0	1	0	0	4[1]
6	1	0	0	0	0	2	1	1	1	0	5.167[2]

Les données du tableau 7 correspondent aux scores attribués par les évaluateurs aux questions liées à l'apprentissage de l'interface. Les évaluateurs ont insisté sur le fait que l'interface ne tient pas compte des utilisateurs expérimentés, et ont suggéré l'ajout d'une sorte de langage de commande. Ceci dans le but d'épargner à un utilisateur la recherche d'une fonctionnalité dans l'un des menus disponibles, du moment qu'il sait que cette fonctionnalité existe et qu'il connait son nom d'appel.

Nous pensons que l'interface MMI^2 incluant tous les modes aurait récolté de meilleurs scores grâce à sa flexibilité par rapport à l'interface graphique de MMI^2 isolée des autres modes de dialogue.

Tableau 8

N°Quest. \ note	0	1	2	3	4	5	6	7	8	9	moy.
1	0	0	1	0	0	0	2	0	2	3	7.125
2	1	0	0	0	0	0	0	3	2	2	6875
3	0	1	0	0	3	0	1	0	2	0	5[1]
4	0	1	0	0	1	2	1	0	2	1	575
5	1	0	1	0	0	1	2	1	1	1	5375
6	0	0	0	1	0	2	3	1	1	0	575
7	0	0	1	1	2	3	0	0	1	0	45

Le tableau 8 récapitule les scores moyens obtenus pour les questions sondant les impressions générales des évaluateurs par rapport à l'interface évaluée.

Résumé

Cette thèse propose une modélisation basée sur la notion de carte planaire. Les données d'un même niveau plan sont modélisées, selon leur sémantique, par des cartes planaires indépendantes qui sont superposées. De même, l'empilement de telles cartes permet de modéliser des données appartenant à des plans strictement parallèles. La visualisation et la manipulation des données de plans strictement parallèles sont assurées par une représentation tri-dimensionnelle inspirée des techniques de dessin en perspective cavalière. Une structure de données hiérarchique offre la possibilité de détailler une zone d'un plan (face d'une carte planaire) par un certain nombre de cartes planaires représentant différentes sémantiques. De plus, cette modélisation gère parfaitement les liens qui peuvent exister entre les objets modélisés dans différentes cartes planaires. Elle permet ainsi de garder le sens global de l'ensemble des objets. Cette modélisation a été utilisée pour réaliser une interface graphique dans le cadre d'un projet européen Esprit MMI^2. Une évaluation ergonomique de cette interface est présentée ainsi que les améliorations apportées à cette interface suite à l'évaluation.

Mots clés

modélisation, cartes planaires, empilement de cartes planaires, hiérarchie de cartes planaires, interface graphique, évaluation ergonomique.

Abstract

This thesis proposes a model based on the planar map notion. Data of the same plane level are modeled by several planar maps each of which corresponding to a class of semantics. These maps are independent and superposed. Modeling of data belonging to parallel planes is also allowed by planar maps superposing. Display and manipulation of data belonging to parallel planes are ensured thanks to a three-dimensional representation that is inspired by perspective techniques in draughtmanship domain. A hierarchical data structure offers the possibility to detail one zone of the plane (a face in planar map notion) by many planar maps corresponding to the different semantics. Furthermore, this model handles perfectly the semantic links that could lie objects modeled in different planar maps. Hence, it preserves the whole meaning of the modeled scene. The proposed model has been used to design a graphical interface within the scope of an european Esprit project MMI^2. An ergonomic evaluation of that interface and refinements made in it as consequence of the evaluation are also presented in this thesis.

Keywords

modeling, planar maps, superposing of planar maps, hierarchy of planar maps, graphical interface, ergonomic evaluation.